Title: Information processing device and method, program and recording medium			
Application Number:	01801575	**Application Date:**	2001.04.2
Publication Number:	1383679	**Publication Date:**	2002.12.0
Approval Pub. Date:	2006.01.25	**Granted Pub. Date:**	2006.01.2
International Classifi-cation:	H04N5/93;G11B20/10		
Applicant(s) Name:	Sony Corp		
Address:			
Inventor(s) Name:	Kato Motoki;Hamada Toshiya		
Attorney & Agent:	ma ying		

Abstract

In case continuous reproduction is commanded from a first AV stream to a second AV stream, a ade up of a preset portion of the first AV stream and a preset portion of the second AV stream, i hird AV stream is reproduced when reproduction is switched from the first AV stream to the second e information pertinent to the third AV stream, the address information of a source packet of the t a timing of switching from the first AV stream to the third AV stream and the address informati ket of the second AV stream at a timing of switching from the third AV stream to the second AV st d. This enables reproduction such as to maintain continuity between separately recorded AV stream

Close

[19] 中华人民共和国国家知识产权局

[51] Int. Cl.
H04N 5/93 (2006.01)
G11B 20/10 (2006.01)

[12] 发明专利说明书

专利号 ZL 01801575.1

[45] 授权公告日 2006 年 1 月 25 日

[11] 授权公告号 CN 1239021C

[22] 申请日 2001.4.20 [21] 申请号 01801575.1

[30] 优先权

[32] 2000. 4.21 [33] JP [31] 183769/00

[32] 2000. 9. 7 [33] JP [31] 271550/00

[86] 国际申请 PCT/JP2001/003417 2001.4.20

[87] 国际公布 WO2001/082610 日 2001.11.1

[85] 进入国家阶段日期 2002.2.1

[71] 专利权人 索尼公司

地址 日本东京都

[72] 发明人 加藤元树 浜田俊也

审查员 沈乐平

[74] 专利代理机构 北京市柳沈律师事务所

代理人 马 莹 邵亚丽

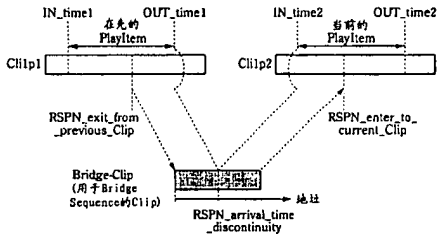
权利要求书 2 页 说明书 55 页 附图 94 页

[54] 发明名称

信息处理设备及方法、程序和记录介质

[57] 摘要

在发布连续再现第一和第二 AV 流的命令时，生成包括第一 AV 流的预定部分和第二 AV 流的预定部分的第三 AV 流，并当第一 AV 流的再现变化到第二 AV 流的再现时再现第三 AV 流。同时生成与第三 AV 流相关的地址信息，所述地址信息包括在将第一 AV 流的再现切换到第三 AV 流的再现时刻关于第一 AV 流的源数据包的地址的信息、以及在将第三 AV 流的再现切换到第二 AV 流的再现时刻关于第二 AV 流的源数据包的地址的信息。从而保持连续再现所分离记录的 AV 流。



1. 一种信息处理设备，包括:

生成装置，用于在命令连续再现从第一 AV 流到第二 AV 流的情况下，生成由所述第一 AV 流的预设部分和所述第二 AV 流的预设部分构成的第三 AV 流、以及作为与所述第三 AV 流相关的信息的地址信息，当再现从所述第一 AV 流切换到所述第二 AV 流时再现所述第三 AV 流，所述地址信息包括在将再现从所述第一 AV 流切换到所述第三 AV 流的时刻所述第一 AV 流的源数据包的地址信息、以及在将再现从所述第三 AV 流切换到所述第二 AV 流的时刻所述第二 AV 流的源数据包的地址信息；以及

记录装置，用于记录由所述生成装置所生成的所述第三 AV 流和所述地址信息。

2. 根据权利要求 1 所述的信息处理设备，其中包括在所述生成装置生成的所述地址信息中的所述第一 AV 流的源数据包的到达时间戳与位于所述第三 AV 流前端的源数据包的到达时间戳彼此连续，而且，包括在所述生成装置生成的所述地址信息中的所述第二 AV 流的源数据包的到达时间戳与位于所述第三 AV 流末端的源数据包的到达时间戳彼此连续。

3. 根据权利要求 2 所述的信息处理设备，其中在所述第三 AV 流的源数据包的到达时间戳中存在唯一不连续点。

4. 根据权利要求 2 所述的信息处理设备，其中，确定所述地址使得在所述生成装置生成的所述地址信息中包含的关于所述第一 AV 流的源数据包地址的信息所指定的源数据包之前的 AV 流的数据部分将被定位在记录介质上不少于预设尺寸的连续区域中。

5. 根据权利要求 2 所述的信息处理设备，其中，确定所述地址使得接在所述生成装置生成的所述地址信息中包含的关于所述第二 AV 流的源数据包地址的信息所指定的源数据包之后的 AV 流的数据部分将被定位在记录介质上不少于预设尺寸的连续区域中。

6. 根据权利要求 2 所述的信息处理设备，其中生成所述第三 AV 流，使得所述第三 AV 流将被定位在所述记录介质上不少于预设尺寸的连续区域中。

7. 一种信息生成方法，包括:

在命令连续再现从第一 AV 流到第二 AV 流的情况下，生成由所述第一

AV 流的预设部分和所述第二 AV 流的预设部分构成的第三 AV 流的步骤，当再现从所述第一 AV 流切换到所述第二 AV 流时再现所述第三 AV 流；以及

生成地址信息的步骤，作为与所述第三 AV 流相关的信息，所述地址信息包括在将再现从所述第一 AV 流切换到所述第三 AV 流的时刻关于所述第一 AV 流的源数据包的地址的信息、以及在将再现从所述第三 AV 流切换到所述第二 AV 流的时刻关于所述第二 AV 流的源数据包的地址的信息。

8. 一种信息处理设备，包括：

再现装置，用于再现其上记录有第一 AV 流、第二 AV 流、第三 AV 流、以及作为与所述第三 AV 流相关的信息的地址信息的记录介质，所述第三 AV 流由所述第一 AV 流的预设部分和所述第二 AV 流的预设部分组成，当再现从所述第一 AV 流切换到所述第二 AV 流时再现所述第三 AV 流，所述地址信息包括在将再现从所述第一 AV 流切换到所述第三 AV 流的时刻关于所述第一 AV 流的源数据包的地址的信息、以及在将再现从所述第三 AV 流切换到所述第二 AV 流的时刻关于所述第二 AV 流的源数据包的地址的信息；以及

控制装置，用于根据读出的与所述第三 AV 流相关的信息，控制所述再现装置将再现从所述第一 AV 流切换到所述第三 AV 流、以及从所述第三 AV 流切换到所述第二 AV 流。

9. 一种信息处理方法，包括：

再现步骤，用于再现其上记录有第一 AV 流、第二 AV 流、第三 AV 流、以及作为与所述第三 AV 流相关的信息的地址信息的记录介质，所述第三 AV 流由所述第一 AV 流的预设部分和所述第二 AV 流的预设部分组成，当再现从所述第一 AV 流切换到所述第二 AV 流时再现所述第三 AV 流，所述地址信息包括在将再现从所述第一 AV 流切换到所述第三 AV 流的时刻关于所述第一 AV 流的源数据包的地址的信息、以及在将再现从所述第三 AV 流切换到所述第二 AV 流的时刻关于所述第二 AV 流的源数据包的地址的信息；以及

控制再现的控制步骤，用于根据读出的与所述第三 AV 流相关的信息，控制将再现从所述第一 AV 流切换到所述第三 AV 流、以及从所述第三 AV 流切换到所述第二 AV 流。

信息处理设备及方法、程序和记录介质

技术领域

本发明涉及信息处理设备及方法、程序和记录介质，更具体地说，涉及构建用于保持再现区间运动画面的连续性的信息处理设备及方法、程序和记录介质。

背景技术

近来已经提出了各种类型的光盘，作为能够从记录设备中移出的记录介质。这些可记录的光盘已经是作为几个 GB 的大容量介质被提出，并且认为有希望作为用于记录诸如视频信号的 AV（视听）信号的介质。在要记录在该可记录光盘上的数字 AV 信号源（供应源）中，有 CS 数字卫星广播和 BS 数字广播。另外，数字系统的地波电视广播也已经为今后使用而提出。

从这些源提供的数字视频信号在 MPEG2（运动图像专家组）系统下被例行地进行画面压缩。在记录设备中，设置了适合于该设备的记录速率。如果数字广播的数字视频信号记录在国内使用的常规画面存储介质中，数字视频信号被首先解码，并且随后进行带宽限制以用于记录。在当然包括 MPEG1 视频、MPEG2 视频和 DV 系统的数字记录系统的情况下，数字视频信号被首先解码，并且随后根据适合于随后记录的设备的记录速率的编码系统进行重新编码。

但是，其中所提供的位流在记录之前一次解码并随后进行带宽限制及重新编码的记录系统有变坏的画面质量。在记录画面压缩数字信号中，如果输入数字信号的传输速率小于记录和/或再现设备的记录速率，直接记录所提供的没有进行解码或者重新编码的位流的方法在画面质量上仅仅在很少程度上遭受破坏。但是，如果输入数字信号的传输速率超过记录和/或再现设备的记录速率，则重新编码位流和记录经过重新编码的位流确实是必须的，结果，在记录和/或再现设备中解码之后，传输速率将不高于盘记录速率的上限。

如果位流在其中输入数字信号的位速率随时间增加或者降低的可变速率系统中传输，则适于在缓冲器中暂时存储数据和突发方式记录该数据的盘记

录设备与具有由旋转头的固定 rpm（每分钟转速）强加的固定记录速率的带式记录系统相比，其记录介质的容量浪费要少。

因此，可以预计，在不远的将来，当数字广播变成主流时，增加对记录和/或再现设备的需求，其中广播信号记录为数字信号，DataStreamer 无需进行解码或重新编码，并且其中盘用作记录介质。

在上述记录设备中再现记录在记录介质上的数据时，存在公知的所谓跳跃再现，其中再现前进到预设的画面，并且接着再现与预设画面相隔开的临时画面。这种跳跃再现具有再现画面暂时失去连续性的缺陷。

发明内容

因此，本发明的目的是在重放区间使再现保持运动画面的连续性。

本发明提供一种信息处理设备，包括：生成装置，用于在命令连续再现从第一 AV 流到第二 AV 流的情况下，生成由第一 AV 流的预设部分和第二 AV 流的预设部分构成的第三 AV 流、以及作为与第三 AV 流相关信息的地址信息，当再现从第一 AV 流切换到第二 AV 流时再现第三 AV 流，所述地址信息包括在将再现从第一 AV 流切换到第三 AV 流的时刻关于第一 AV 流的源数据包的地址的信息、以及在将再现从第三 AV 流切换到第二 AV 流的时刻关于第二 AV 流的源数据包的地址的信息；以及记录装置，用于记录由生成装置所生成的第三 AV 流和地址信息。

最好，包括在生成装置所生成的地址信息中的第一 AV 流的源数据包的到达时间戳、以及位于第三 AV 流前端的源数据包的到达时间戳彼此连续，并且，包括在生成装置所生成的地址信息中的第二 AV 流的源数据包的到达时间戳、以及位于第三 AV 流末端的源数据包的到达时间戳彼此连续。

最好，在第三 AV 流的源数据包的到达时间戳中存在唯一不连续点。

最好，确定地址使得在生成装置生成的地址信息中包含的关于第一 AV 流的源数据包地址的信息所指定的源数据包之前的 AV 流的数据部分将被定位在记录介质上不少于预设尺寸的连续区域中。

最好，确定地址使得接在生成装置生成的地址信息中包含的关于第二 AV 流的源数据包地址的信息所指定的源数据包之后的 AV 流的数据部分将被定位在记录介质上不少于预设尺寸的连续区域中。

最好，生成第三 AV 流，使得第三 AV 流将被定位在记录介质上不少于

预设尺寸的连续区域中。

本发明还提供一种信息生成方法，包括：在命令连续再现从第一 AV 流到第二 AV 流的情况下，生成由第一 AV 流的预设部分和第二 AV 流的预设部分构成的第三 AV 流的步骤，当再现从第一 AV 流切换到第二 AV 流时再现第三 AV 流；以及生成作为与第三 AV 流相关的信息的地址信息的步骤，所述地址信息包括在将再现从第一 AV 流切换到第三 AV 流的时刻关于第一 AV 流的源数据包的地址的信息、以及在将再现从第三 AV 流切换到第二 AV 流的时刻关于第二 AV 流的源数据包的地址的信息。

本发明还提供一种记录介质，其上记录有计算机可读程序，其中该程序包括：在命令连续再现从第一 AV 流到第二 AV 流的情况下，生成由第一 AV 流的预设部分和第二 AV 流的预设部分构成的第三 AV 流的步骤，当再现从第一 AV 流切换到第二 AV 流时再现第三 AV 流；以及生成作为与第三 AV 流相关的信息的地址信息的步骤，所述地址信息包括在将再现从第一 AV 流切换到第三 AV 流的时刻关于第一 AV 流的源数据包的地址的信息、以及在将再现从第三 AV 流切换到第二 AV 流的时刻关于第二 AV 流的源数据包的地址的信息。

本发明还提供一种用于使计算机执行程序的程序，其中该程序包括：在命令连续再现从第一 AV 流到第二 AV 流的情况下，生成由第一 AV 流的预设部分和第二 AV 流的预设部分构成的第三 AV 流的步骤，当再现从第一 AV 流切换到第二 AV 流时再现第三 AV 流；以及生成作为与第三 AV 流相关的信息的地址信息的步骤，所述地址信息包括在将再现从第一 AV 流切换到第三 AV 流的时刻关于第一 AV 流的源数据包的地址的信息、以及在将再现从第三 AV 流切换到第二 AV 流的时刻关于第二 AV 流的源数据包的地址的信息。

本发明还提供一种信息处理设备，包括：第一读出装置，用于从记录介质上读出第一 AV 流、第二 AV 流、或第三 AV 流；第二读出装置，用于读出关于在将再现从第一 AV 流切换到第三 AV 流的时刻关于第一 AV 流的源数据包的地址信息、以及关于在将再现从第三 AV 流切换到第二 AV 流的时刻关于第二 AV 流的源数据包的地址信息，作为与第三 AV 流相关的信息；以及再现装置，用于根据由第二读出装置读出的与第三 AV 流相关的信息，当将再现从第一读出装置读出的第一 AV 流切换到第三 AV 流、并从第三 AV 流切换到第二 AV 流时，执行再现。

本发明还提供一种信息处理方法，包括：第一读出控制步骤，用于从记录介质上读出第一 AV 流、第二 AV 流、或第三 AV 流；第二读出控制步骤，用于读出在将再现从第一 AV 流切换到第三 AV 流的时刻关于第一 AV 流的源数据包的地址的信息、以及在将再现从第三 AV 流切换到第二 AV 流的时刻关于第二 AV 流的源数据包的地址的信息，作为与第三 AV 流相关的信息；以及再现步骤，当由第二读出控制步骤读出的与第三 AV 流相关的信息，用于执行将再现从第一读出控制步骤读出的第一 AV 流切换到第三 AV 流、并从第三 AV 流切换到第二 AV 流时，执行再现。

本发明还提供一种记录介质，其上记录有计算机可读程序，其中该程序包括：第一读出控制步骤，用于从记录介质上读出第一 AV 流、第二 AV 流、或第三 AV 流；第二读出控制步骤，用于读出在将再现从第一 AV 流切换到第三 AV 流的时刻关于第一 AV 流的源数据包的地址的信息、以及在将再现从第三 AV 流切换到第二 AV 流的时刻关于第二 AV 流的源数据包的地址的信息，作为与第三 AV 流相关的信息；以及再现步骤，用于根据由第二读出控制步骤读出的与第三 AV 流相关的信息，当将再现从第一读出控制步骤读出的第一 AV 流切换到第三 AV 流、并从第三 AV 流切换到第二 AV 流时，执行再现。

本发明还提供一种用于使计算机执行程序的程序，其中该程序包括：第一读出控制步骤，用于从记录介质上读出第一 AV 流、第二 AV 流、或第三 AV 流；第二读出控制步骤，用于读出在将再现从第一 AV 流切换到第三 AV 流的时刻关于第一 AV 流的源数据包的地址的信息、以及在将再现从第三 AV 流切换到第二 AV 流的时刻关于第二 AV 流的源数据包的地址的信息，作为与第三 AV 流相关的信息；以及再现步骤，用于根据由第二读出控制步骤读出的与第三 AV 流相关的信息，当将再现从第一读出控制步骤读出的第一 AV 流切换到第三 AV 流、并从第三 AV 流切换到第二 AV 流时，执行再现。

本发明还提供一种其上记录有地址信息的记录介质，包括：在命令连续再现从第一 AV 流到第二 AV 流的情况下，由第一 AV 流的预设部分和第二 AV 流的预设部分构成的第三 AV 流，当再现从第一 AV 流切换到第二 AV 流时再现第三 AV 流；以及地址信息，作为与第三 AV 流相关的信息，所述地址信息包括在将再现从第一 AV 流切换到第三 AV 流的时刻关于第一 AV 流的源数据包的地址的信息、以及在将再现从第三 AV 流切换到第二 AV 流的时刻关于第二 AV 流的源数据包的地址的信息。

在根据本发明的信息处理设备、方法及程序中，如果命令执行从第一 AV 流到第二 AV 流的连续再现，则生成由第一 AV 流的预设部分和第二 AV 流的预设部分构成的第三 AV 流，并且当再现从第一 AV 流切换到第二 AV 流时再现第三 AV 流；同时生成地址信息，作为与第三 AV 流相关的信息，所述地址信息包括在将再现从第一 AV 流切换到第三 AV 流的时刻关于第一 AV 流的源数据包的地址的信息、以及在将再现从第三 AV 流切换到第二 AV 流的时刻关于第二 AV 流的源数据包的地址的信息。

在根据本发明的信息处理设备、方法及程序中，从记录介质上读出第一 AV 流、第二 AV 流、或第三 AV 流；并从记录介质上读出地址信息作为与第三 AV 流相关的信息，所述地址信息包括在将再现从第一 AV 流切换到第三 AV 流的时刻关于第一 AV 流的源数据包的地址的信息、以及在将再现从第三 AV 流切换到第二 AV 流的时刻关于第二 AV 流的源数据包的地址的信息；以及根据读出的与第三 AV 流相关的信息，在再现进行时，将再现从第一 AV 流切换到第三 AV 流、并从第三 AV 流切换到第二 AV 流。

通过下面结合附图对本发明实施例的说明，本发明的其它目的、特征和优点将变得更加清楚。

附图简要说明

图 1 表示根据本发明记录和/或再现设备的实施例的结构；

图 2 表示在记录介质上通过记录和/或再现设备 1 记录的数据的数据格式；

图 3 表示实 PlayList 和虚 PlayList；

图 4A，4B 和 4C 表示实 PlayList 的创建；

图 5A，5B 和 5C 表示实 PlayList 的删除；

图 6A 和 6B 表示汇编编辑；

图 7 表示虚 PlayList 中提供的子路径；

图 8 表示 PlayList 重放序列的改变；

图 9 表示 PlayList 上的标记和 Clip 上的标记（mark）；

图 10 表示菜单缩略图（menu thumbnail）；

图 11 表示加到 PlayList 的标记；

图 12 表示加到 Clip 的标记；

图 13 表示在 PlayList、Clip 和缩略图文件之间的关系;

图 14 表示目录结构;

图 15 表示 infr.dvr 的句法;

图 16 表示 DVRVolume 的句法;

图 17 表示 ResumeVolume 的句法;

图 18 表示 UIAppInfoVolume 的句法;

图 19 表示字符集值表;

图 20 表示 TableOfPlayList 的句法;

图 21 表示 TableOfPlayList 的另一个句法;

图 22 表示 MakersPrivateData 的句法;

图 23 表示 xxxx.rpls 和 yyyy.vpls 的句法;

图 24A 到 24C 表示 PlayList;

图 25 表示 PlayList 的句法;

图 26 表示 PlayList_type 表;

图 27 表示 UIAppInfoPlayList 的句法;

图 28A 到 28C 表示图 27 所示的 UIAppInfoPlayList 句法中的标志(flag);

图 29 表示 PlayItem;

图 30 表示 PlayItem;

图 31 表示 PlayItem;

图 32 表示 PlayItem 的句法;

图 33 表示 IN_time;

图 34 表示 OUT_time;

图 35 表示 Connection_Condition 表;

图 36A 到 36D 表示 Connection_Condition;

图 37 表示 BridgeSequenceInfo;

图 38 表示 BridgeSequenceInfo 的句法;

图 39 表示 SubPlayItem;

图 40 表示 SubPlayItem 的句法;

图 41 表示 Mark_type 表;

图 42 表示 PlayListMark 的句法;

图 43 表示 Mark_type 表;

图 44 表示 Mark_time_stamp;

图 45 表示 zzzzz.clip 的句法;

图 46 表示 ClipInfo 的句法;

图 47 表示 Clip_stream_type 的表;

图 48 表示 offset_SPN;

图 49 表示 offset_SPN;

图 50A、50B 表示 STC 域;

图 51 表示 STC_Info;

图 52 表示 STC_Info 的句法;

图 53 表示 ProgramInfo;

图 54 表示 ProgramInfo 的句法;

图 55 表示 VideoCondingInfo 的句法;

图 56 表示 Video_format 的表;

图 57 表示 frame_rate 的表;

图 58 表示 display_aspect_ratio 的表;

图 59 表示 AudioCondingInfo 的句法;

图 60 表示 audio_coding 的表;

图 61 表示 audio_component_type 的表;

图 62 表示 sampling_frequency 的表;

图 63 表示 CPI;

图 64 表示 CPI;

图 65 表示 CPI 的句法;

图 66 表示 CPI_type 的表;

图 67 表示视频 EP_map;

图 68 表示 EP_map;

图 69 表示 EP_map;

图 70 表示 EP_map 的句法;

图 71 表示 EP_typevalues 的表;

图 72 表示 EP_map_for_one_stream_PID 的句法;

图 73 表示 TU_map;

图 74 表示 TU_map 的句法;

图 75 表示 ClipMark 的句法;
图 76 表示 Mark_type 的表;
图 77 表示 Mark_type_stamp 的表;
图 78 示出 menu.thmb 和 mark.thmb 的句法;
图 79 示出 thumbnail 的句法;
图 80 示出 thumbnail_picture_format 表;
图 81A 和 81B 示出 tn_block;
图 82 示出 DVR MPEG2 传输流的结构;
图 83 示出 DVR MPEG2 传输流的记录器模型;
图 84 示出 DVR MPEG2 传输流的播放器模型;
图 85 示出源数据包的句法;
图 86 示出 TP_extra_header 的句法;
图 87 示出允许复制指示符的表;
图 88 示出无缝连接;
图 89 示出无缝连接;
图 90 示出无缝连接;
图 91 示出无缝连接;
图 92 示出无缝连接;
图 93 示出音频交迭;
图 94 示出使用 BridgeSequence 的无缝连接;
图 95 示出不使用 BridgeSequence 的无缝连接;
图 96 示出 DVR STD 模型;
图 97 示出用于解码和显示的时序图;
图 98 是用于说明准备 RealPlayList 的流程图;
图 99 是用于说明定制 VirtualPlayList 的流程图;
图 100 是用于说明定制 Bridge Sequence 的流程图;
图 101 是用于说明再现 PlayList 的流程图;
图 102 示出介质。

实施本发明的最佳模式

引用附图，现在详细地说明本发明的实施例。图 1 表示体现本发明的记

录和/或再现设备 1 的典型内部结构。首先，说明构造为记录从外部输入的信号的记录单元 2 的结构。记录和/或再现设备 1 构造为用模拟或者数字数据供给和记录模拟或数字数据。

模拟视频信号和模拟音频信号分别提供给端子 11，12。输入到端子 11 的视频信号输出到分析单元 14 和 AV 编码器 15。输入到端子 12 的音频信号输出到分析单元 14 和 AV 编码器 15。分析单元 14 从输入的视频和音频信号中提取诸如场景变化的特征点。

AV 编码器 15 编码输入的视频和音频信号以将诸如编码视频流（V）、编码音频流（A）和 AV 同步的系统信息输出给多路复用器 16。

编码视频流是例如用 MPEG（运动图像专家组）2 系统编码的视频流，而编码音频流是根据 MPEG1 系统编码的音频流，编码音频流是在例如 MPEG1 系统中编码的音频流或者根据杜比 AC3（商标）系统编码的音频流。多路复用器 16 基于输入系统信息多路复用输入视频流和音频流以通过开关 17 将多路复用流输出到多路复用的流分析单元 18 和源打包器（source packetizer）19。

多路复用流例如是 MPEG-2 传输流或者 MPEG-2 节目流。根据在其上记录该流的记录介质 100 的应用格式，源打包器 19 将输入的多路复用流编码成由源数据包构成的 AV 流。在输出到写单元 22 之前，AV 流在 ECC（错误校正和编码）单元 20 和调制单元 22 中用附带的 ECC 代码和调制进行处理，其然后基于由控制器 23 输出的控制信号写（记录）AV 流。

从数字接口或者数字电视调谐器输入的诸如数字电视广播的传输流输入到端子 13。有两个用于记录输入到端子 13 的传输流的记录系统：一个是透明记录系统，另一个是记录之前是重新编码的系统，重新编码的目的在于降低例如记录位速率。记录系统命令信息从作为用户接口的端子 24 输入到控制

器 23。

在输入的传输流的透明记录中，输入到端子 13 的传输流通过开关 17 输出到多路复用流分析单元 18 和源打包器 19。如上述，在记录介质上记录 AV 流的随后处理与编码和记录模拟输入的音频和视频信号的随后处理相同，因此，为了简单起见在此不进行说明。

如果输入的传输流被重新编码和随后记录，则输入到端子 13 的传输流馈送到多路分用器 26，它多路分用输入的传输流以提取视频流（V）、音频流（A）和系统信息（S）。

在通过多路分用器 26 提取的流（信息）中，视频流输出到音频解码器 27，而音频流和系统信息输出到多路复用器 16。音频解码器 27 解码输入的传输流以将编码的视频流（V）输出到多路复用器 16。

从多路分用器 26 输出的且输入到多路复用器 16 的音频流和系统信息以及由 AV 编码器 15 输出的视频流基于输入系统信息被多路复用，并通过开关 17 作为多路复用流输出到多路复用流分析单元 18 和源打包器 19。如上述，在记录介质上记录 AV 流的随后处理是与编码和记录模拟输入的音频和视频信号的随后处理相同，因此，为了简单起见在此不进行说明。

本实施例的记录和/或再现设备 1 在记录介质 100 上记录 AV 流文件，同时也记录解释该文件的应用数据库信息。对控制器 23 的输入信息是来自分析单元 14 的运动画面的特征信息，来自多路复用流分析单元 18 的 AV 流的特征信息和从端子 24 输入的用户命令信息。

在 AV 编码器 15 编码视频信号时，从分析单元 14 提供的运动画面的特征信息是由分析单元 14 产生的。分析单元 14 分析输入视频和音频信号的内容以产生与输入的运动画面信号的画面特征（Clip 标记）有关的信息。该信息是表示诸如节目开始点、场景变化点、CM 商业广告开始和结束点、输入视频信号中的标题或者幻灯机（telop）的特征 Clip 标记点的画面的信息，并且还包括与音频信号的立体音/非立体音转换点和消音部分有关的画面和信息的缩略图（thumbnail）。

上述画面表示信息通过控制 23 馈送到多路复用器 16。当多路复用由控制器 23 指定为 Clip 标记的编码画面时，多路复用器 16 将用于指定 AV 流上编码画面的信息返回到控制器 23。具体地，该信息是画面的 PTS（显示时间戳）或者是画面编码版本的 AV 流上的地址信息。控制器 23 存储特征画面的

分类和用于指定 AV 流上相互关联的编码画面的信息。

来自多路复用流分析单元 18 的 AV 流的特征信息是与要记录的 AV 流编码信息相关的信息，并且通过分析单元 18 记录。例如，特征信息包括 AV 流中 I 画面的时间戳（time stamp）和地址信息、系统时钟的非连续点信息、AV 流的编码参数和 AV 流中编码参数的改变点信息。当透明地记录从端子 13 输入的传输流时，多路复用流分析单元 18 从输入传输流中检测前述的 Clip 标记的画面，并且产生用于指定由 Clip 标记和其类型指定的画面的信息。

来自端子 24 的用户指配信息是指定由用户指定的重放域的信息、用于解释重放域内容的字符字母、或者诸如书签或者由用户为他或她喜爱场景而设定的重新开始点的信息。

基于前述输入信息，控制器 23 创建 AV 流数据库（Clip），AV 流重放域（PlayItem）的组（PlayList）的数据库，记录介质 100 记录内容的管理信息（info.dvr）和关于缩略图画面的信息。类似于 AV 流，从上述信息构成的应用数据库信息在 ECC 单元 20 和调制单元 21 中进行处理且输入到写单元 22，其然后将数据库文件记录在记录介质 100 上。

随后将详细说明上述的应用数据库信息。

当在记录介质 100 上记录的 AV 流文件（画面数据和语音数据文件）和因此在记录介质 100 上记录的应用数据库信息通过再现单元 3 再现时，控制器 23 首先命令读出单元 28 从记录介质 100 中读出应用数据库信息。读出单元 28 从记录介质 100 中读出应用数据库信息，然后从记录介质 100 中读出应用数据库信息以通过由解调单元 29 和 ECC 解码器 30 进行的解调和错误校正处理来将该应用数据库信息发送到控制器 23。

基于应用数据库信息，控制器 23 将在记录介质 100 上记录的 PlayList 表输出到端子 24 的用户接口。用户从 PlayList 表中选择希望再现的 PlayList。指定为要再现的与 PlayList 相关的信息输入到控制器 23。控制器 23 命令读出单元 28 读出在再现 PlayList 中必需的 AV 流文件。根据该命令，读出单元 28 从记录介质 100 中读出对应的 AV 流以将所读出的 AV 流输出到解调单元 29。因此，输入到解调单元 29 的 AV 流通过预置处理被解调和通过 ECC 解码器 30 的处理输出到源解数据包器（depacketizer）31。

源解数据包器 31 将从记录介质 100 读出的并以预置方式处理的应用格式的 AV 流变换成可由多路分用器 26 处理的流。多路分用器 26 将形成由控制

器 23 指定的 AV 流的重放域（PlayItem）的诸如视频流（V）、音频流（A）或者 AV 同步的系统信息（S）输出到音频解码器 27，该 AV 解码器 27 解码视频流和音频流以将重放视频信号和重放音频信号分别输出到相关的端子 32、33。

如果从作为用户接口的端子 24 提供指令进行随机存储重放或者指定重放的信息，则控制器 23 基于 AV 流数据库（Clip）内容确定来自记录介质 100 的 AV 流的读出位置，以命令读出单元 28 读出 AV 流。如果作为用户选择的 PlayList 将作为从预置时间点进行再现，则控制器 23 命令读出单元 28 从具有最靠近指定的时间点的时间戳的 I 画面读出数据。

当用户从存储在 Clip 信息的 ClipMark 中的节目的索引点或者场景变化点已经选择了某一 Clip 标记，即正如作为用户接口所显示的，当用户从存储在 ClipMark 中的索引点或者场景变化点的缩略图画面表中选择了某一画面时，则控制器 23 确定来自记录介质 100 之 AV 流读出位置，以命令读出单元 28 读出 AV 流。即，控制器 23 命令读出单元 28 从具有最靠近已经存储用户选择的画面的 AV 流地址的地址的 I 画面中读出数据。读出单元 28 从指定地址读出数据。读出的数据通过解调单元 29、ECC 解码器 30 和通过源打包器 19 处理以便提供到多路分用器 26，并且通过音频解码器 27 解码以再现由标记点画面的地址表示的 AV 数据。

如果用户已经命令快进（fast forward）重放，则控制器 23 命令读出单元 28 基于 AV 流数据库（Clip）依次连续地读出 AV 流中的 I 画面数据。

读出单元 28 从指定的随机存取点中读出 AV 流的数据。如此读出的数据通过由下游侧的各种元件的处理再现。

现在说明其中用户编辑记录在记录介质 100 上的 AV 流的情况。如果希望指定的记录在记录介质 100 上的 AV 流的重放域，例如，如果希望创建从歌曲节目 A 中再现由歌唱者 A 演唱部分并且随后从另一个歌曲节目 B 中再现由同一歌唱者 A 演唱部分的重放例程，则与重放域的 IN_point（开始点）和 OUT_point(结束点)相联系的信息从作为用户接口的端子输入到控制器 23。控制器 23 创建 AV 流重放域（PlayItem）的组的数据库（PlayList）。

当用户希望擦除记录在记录介质 100 上的一部分 AV 流时，与擦除域的 IN_point 和 OUT_point 相关的信息输入到控制器 23，其然后改进 PlayList 数据库以便仅仅参照需要的 AV 流。控制器 23 还命令写单元 22 擦除 AV 流的不

需要的流部分。

现在说明这种情况，其中用户希望指定记录在记录介质上的 AV 流的重放域以创建新的重放例程（route）和以无缝方式互连对应的重放域。在这种情况下，控制器 23 创建 AV 流重放域（PlayItem）的组的数据库（PlayList）并进行部分地重新编码与重新多路复用在重放域连接点附近的视频流。

在重放域 IN_point 的画面信息和 OUT_point 的画面信息从端子 24 输入到控制器 23。控制器 23 命令读出单元 28 读出所需要的在 IN_point 和在 OUT_point 上再现的画面数据。读出单元 28 从记录介质 100 上读出数据。如此读出的数据通过解调单元 29、ECC 解码器 30 和源打包器 19 输出到多路分用器 26。

控制器 23 分析输入到多路分用器 26 的数据以确定对视频流的重新编码方法（picture_coding_type 的变化和重新编码之编码位数量的分配）和重新多路复用系统，以将该系统送到 AV 编码器 15 和多路复用器 16。

多路分用器 26 然后将输入的流分离成视频流（V）、音频流（A）和系统信息（S）。视频流可以分类成输入到音频解码器 27 的数据和输入到多路复用器 16 的数据。前者是重新编码所需要的数据，并且通过音频解码器 27 解码，其中具有然后由 AV 编码器 15 重新编码和由此变成视频流的解码画面。后者数据是从没有重新编码的原始流中拷贝的数据。音频流和系统信息直接输入到多路复用器 16。

多路复用器 16 基于从控制器 23 输入的信息多路复用输入流以输出该多路复用流，其被 ECC 单元 20 和调制单元 21 处理以便发送到写单元 22。写单元 22 基于从控制器 23 提供的控制信号将 AV 流记录在记录介质 100 上。

下面解释应用数据库信息和基于该信息的诸如重放和编辑的操作。图 2 表示用于 AV 流管理的应用格式结构，该结构具有两层，即 PlayList 和 Clip。卷（volume）信息管理盘中的所有 Clips 和 PlayList。这里，成对的一个 AV 流和其辅助信息认为是一个对象，并叫作 Clip。AV 流文件叫作 Clip AV 流文件，具有叫作 Clip 信息文件的辅助信息。

一个 Clip AV 流文件存储对应于配置成由应用格式指定的结构的 MPEG 2 传输流的数据。一般说来，文件作为字节串进行处理。Clip AV 流文件的内容在时间轴上扩展，其中具有主要以时间基指定的 Clip（I 画面）入口点。当给定对预置 Clip 的存取点的时间戳时，Clip 信息文件在找到开始在 Clip AV

流文件中读出的数据的地址信息方面是有用的。

参照图 3，现在解释 PlayList，其是为用户从 Clip 中选择希望观看的重放域和方便地编辑重放域而提供的。一个 PlayList 是 Clip 中的一组重放域。预置 Clip 中的一个重放域叫作 PlayItem 并且由时间轴上的一对 IN_point 和 OUT_point 表示。这样，PlayList 是由一组复合 PlayItem 形成的。

PlayList 分类成两个类型，一种是实 PlayList，另一种是虚 PlayList。实 PlayList 共同拥有其引用的 Clip 流部分。即，实 PlayList 在盘上占据对应于其引用的 Clip 流部分之数据容量，并且当擦除实 PlayList 时，其引用的 Clip 流部分的数据也被擦除。

虚 PlayList 不是共同拥有 Clip 数据。因此，如果虚 PlayList 改变或者被删除，而 Clip 的内容不会改变。

解释实 PlayList 的编辑。图 4A 表示实 PlayList 的创建，并且如果 AV 流作为新的 Clip 记录，则引用整个 Clip 的实 PlayList 是新的创建操作。

图 4B 表示实 PlayList 的划分，即在希望点上划分实 PlayList 的操作以将实 PlayList 分成两个实 PlayList。当两个节目在通过单个 PlayList 管理的一个 Clip 管理时，并且当用户趋于重新寄存或者重新记录该节目作为单个独立节目时，进行该划分操作。该操作不会导致 Clip 内容的改变，即对 Clip 本身的划分。

图 4C 表示将两个实 PlayList 组合成一个新的实 PlayList 之操作的实 PlayList 的组合操作。该组合操作是当诸如用户希望重新寄存两个节目为单个程序时完成的。该操作不会导致 Clip 内容的改变，即是将 Clip 本身组合为一个。

图 5A 表示整个实 PlayList 的删除。如果删除整个预置的实 PlayList 的操作，则由所删除的实 PlayList 引用的 Clip 的相关流部分也被删除。

图 5B 表示实 PlayList 的部分删除。如果删除实 PlayList 的期望部分，则相关 PlayItem 改变为仅仅引用所需要的 Clip 流部分。对应的 Clip 流部分被删除。

图 5C 表示实 PlayList 的最小化。其是使与实 PlayList 相关的 PlayItem 仅仅引用虚 PlayList 需要的 Clip 流部分的操作。对应的虚 PlayList 不需要的 Clip 流部分被删除。

如果实 PlayList 通过上述操作改变使得由实 PlayList 引用的 Clip 流部分

被删除，则有这种可能性，即出现采用所删除的 Clip 的虚 PlayList，使得在虚 PlayList 中可以产生因所删除的 Clip 导致的问题。

为了防止这种情况发生，则向用户显示这种消息："如果存在引用实 PlayList 正在引用的 Clip 流部分的虚 PlayList，且实 PlayList 被删除，则虚 PlayList 本身被删除——可以否？"，通过证实或者警告响应用户的删除操作，此后，执行删除处理，或者取消用户命令的对象。或者，完成实 PlayList 的最小化操作以代替删除虚 PlayList。

现在解释虚 PlayList 的操作。如果操作是对虚 PlayList 进行的，则 Clip 的内容不改变。图 6A 和 6B 表示汇编和编辑（IN-OUT 编辑）。这是创建用户希望观看的重放域的 PlayItem 以创建虚 PlayList 的操作。在 PlayItem 之间的无缝连接是由应用格式支持的，如后述。

如果存在两个实 PlayList1、2 和与对应实 PlayList 相联系的 Clip1、2，则用户指定实 PlayList1 中的预置域（从 IN1 到 OUT1 的域：PlayItem1）为重放域，并且作为下一个要显示的域，还指定实 PlayList2 中的预置域（从 IN2 到 OUT2 的域：PlayItem2）为重放域，如图 6A 所示。准备了由 PlayItem1 和 PlayItem2 组成的单个虚 PlayList，如图 6B 所示。

现在解释虚 PlayList 的重新编辑。该重新编辑可以通过交替虚 PlayList 中的 IN-或者 OUT 点将新 PlayItem 插入或者附加到虚 PlayList 以及删除虚 PlayList 中的 PlayItem 来列举。虚 PlayList 本身也可以被删除。

图 7 表示对虚 PlayList 的音频转录（后记录）。其是将音频后记录寄存到虚 PlayList 作为子路径的操作。该音频后记录由应用软件支持。附加音频流作为子路径被加到虚 PlayList 主路径的 AV 流。

实 PlayList 和虚 PlayList 的共同点是图 8 所示的改变（移动）PlayList 重放顺序的操作。该操作是盘（卷）中 PlayList 重放顺序的改变并且由在应用格式中定义的 TableOfPlayList 所支持，正如下面引用例如图 20 要说明的。该操作不会导致 Clip 内容的改变。

现在解释标记（Mark）。标记是为指定 Clip 和 PlayList 中的加亮或者特征时间而提供的，如图 9 所示。加到 Clip 的标记叫做 ClipMark。ClipMark 是例如节目索引点或者场景变化点，用于指定起因于 AV 流中内容的特征场景。ClipMark 是由例如图 1 分析单元 14 产生的。当 PlayList 再现时，可以引用和使用由 PlayList 引用的 Clip 的标记。

附加到 PlayList 的标记叫做 PlayListMark（播放表标记）。PlayListMark 是例如由用户设置的书签点或者恢复点（resume point）。对 Clip 和对 PlayList 的标记的设置是将表示标记时间点的时间戳加到标记表。另一方面，标记删除就是从标记表中移去标记的时间戳。结果，AV 流不会由标记设置或者标记删除所改变。

作为 ClipMark 的另一个格式，由 ClipMark 引用的画面可以以 AV 流中的地址基来指定。Clip 上的标记设置就是将表示标记点的画面的地址基信息加到标记表中。另一方面，标记删除就是从标记表中移去表示标记点画面的地址基信息。结果，AV 流不会由标记设置或者标记删除所改变。

现在解释缩略图。缩略图是加到 Volume（卷）、PlayList 和 Clip 的静止画面。有两种类型的缩略图，其中之一是作为表示内容的代表画面的缩略图。这主要是用在主画面中，为的是让用户用光标（未示出）选择他或她希望观看的内容。另一种缩略图是表示由标记点指向的场景的画面。

Volume 和对应 PlayList 需要具有代表画面。当盘设置在记录和/或再现设备 1 中的位置时，Volume 的代表画面被预先提出用作初始地展示表示盘内容的静止画面。注意，盘的意思是预先提出作为盘形状的记录介质 100。PlayList 的代表画面被预先提出用作表示 PlayList 内容的静止画面。

作为 PlayList 的代表画面，可以考虑使用 PlayList 的初始画面作为缩略图（代表画面）。但是，在 0 重放时间的引导画面不必要是表示内容的最佳画面。因此，允许用户设置可选择性画面作为 PlayList 的缩略图。两种类型的缩略图，即作为表示 Volume 之代表画面的缩略图和作为表示 PlayList 之代表画面的缩略图，叫作菜单缩略图。由于经常显示菜单缩略图，这些缩略图需要从盘中以升高的速度读出。因此，以单个文件存储该全部菜单缩略图是有效率的。菜单缩略图不必是从卷的运动画面中提取出的画面，但是可以是从个人计算机或者是数字静止摄像机获取的画面，如图 10 所示。

另一方面，Clip 和 PlayList 需要用复合标记来做标记，同时标记点的画面需要被容易地观看，目的是获得标记位置的内容。表示这种标记点的画面叫作标记缩略图。因此，作为标记操作的正本的画面主要是所提取的标记点画面而不是从外部获取的画面。

图 11 表示附加到 PlayList 的标记和标记缩略图之间的关系，同时图 12 表示附加到 Clip 的标记和标记缩略图之间的关系。与菜单缩略图的区别，标

记缩略图用在例如用于表示 PlayList 细节的子菜单中，而其不要求在短的存取时间中读出。所以，无论何时要求缩略图，记录和/或再现设备 1 打开文件和读出一部分该文件，同时即使由记录和/或再现设备 1 进行的文件打开和读出一部分文件要占用一些时间，也不会出现任何问题。

为了减少在卷中出现的文件数，最好将整个标记缩略图存储在一个文件中。尽管 PlayList 可以具有一个菜单缩略图和多个标记缩略图，但不要求用户直接地选择 Clip（通常，Clip 是通过 PlayList 选择的），因此没有必要提供菜单缩略图。

图 13 表示菜单缩略图，标记缩略图，PlayList 和 Clip 之间的关系。在菜单缩略图中，文件归档为从一个 PlayList 向另一个提供的菜单缩略图。在菜单缩略图中，文件包含有卷缩略图，其表示在盘上记录的数据内容。在菜单缩略图中，文件归档为从一个 PlayList 到另一个和从一个 Clip 到另一个创建的缩略图。

下面说明 CPI（特征点信息）。CPI 是包含在 Clip 信息文件中的数据，并且主要用于发现 Clip AV 流文件中的数据地址，在该地址，当提供 Clip 存取点的时间戳时开始数据读出。在本实施例中使用两种类型的 CPI，其中之一是 EP_map，另一种是 TU_map。

EP_map 是从基本流和传输流中提取的入口点（entry point）（EP）数据的表。其具有用来发现在此开始解码的 AV 流中入口点地点的地址信息。一个 EP 数据由显示时间戳（PTS）和与 PTS 相联系的存取单元的 AV 流中的数据地址构成的，该数据地址与 PTS 配对。

EP_map 主要用于两个目的。第一，其用于发现在由 PlayList 的 PTS 引用的存取单元中的 AV 流的数据地址。第二，EP_map 用于快速前进重放或者快速后退重放。在通过记录和/或再现设备 1 记录输入 AV 流时，如果流的句法能够被分析，则在盘上创建和记录 EP_map。

TU_map 具有从通过数字接口输入的传输数据包的到达时间点得出的时间单元（TU）数据的表。其提供了基于到达时间的时间 arrival_time_based 和 AV 流中数据地址之间的关系。当记录和/或再现设备 1 记录输入 AV 流并且流的句法不能被分析时，则在盘上创建和记录 TU_map。

STCInfo 存储 AV 流文件中的不连续点信息，该 AV 流文件存储了 MPEG-2 传输流。

当 AV 流具有 STC 的不连续点时，相同的 PTS 值可以出现在 AV 流文件中。因此，如果 AV 流中的时间点是根据 PTS 基指定的，则存取点的 PTS 不足以指定该点。而且，还要求包含 PTS 的连续 STC 域的索引。在该格式中，连续 STC 域和其索引被分别叫做 STC 序列和 STC_sequence_id(STC 序列 id)。STC 序列信息是由 Clip 信息文件的 STCInfo 定义的。

STC_sequence_id 用在 AV 流文件中并且在具有 TU_map 的 AV 流文件中是可选择的。

节目是每个基本流的集合并且共同拥有对这些流进行同步再现的单个系统时间基。

再现设备（图 1 的记录和/或再现设备 1）在其解码之前知道 AV 流的内容是有用的。这些内容包括例如传输音频或者视频基本流的传输数据包的 PID 值，或者诸如 HDTV 视频或者 MPEG-2 AAC 音频流的视频或者音频元件的类型。该信息对创建用于给用户显示引用 AV 流的 PlayList 内容的屏幕菜单是有用的。其对于设置对应设备之 AV 解码器和多路分用器的初始状态是同样有用的。

由于这个原因，Clip 信息文件拥有用于说明节目内容的 ProgramInfo。

可以发生的是，节目内容在其中存储了 MPEG-2 传输流的 AV 流文件中应当可以改变。例如，可以改变传输视频基本流的传输数据包的 PID，或者可以将视频流的元件类型从 SDTV 改变为 HDTV。

ProgramInfo 存储了关于 AV 流文件中节目内容的改变点的信息。其中节目内容保持不变的 AV 流文件的域叫作 program_sequence（节目序列）。

该节目序列用在具有 EP_map 的 AV 流文件中，并且在具有 TU_map 的 AV 流文件中是可选择的。

本实施例定义了自身编码流格式（SESF）。该 SESF 用于编码模拟输入信号和用于解码数字输入信号，其随后用于顺序地将解码的信号编码成 MPEG-2 传输流。

SESF 定义了与 MPEG-2 传输流和 AV 流有关的基本流。当记录和/或再现设备 1 编码和记录 SESF 流时，在盘上创建和记录 EP_map。

数字广播流使用用于在记录介质 100 进行记录的下述系统之一：第一，数字广播流自动解码成 SESF 流。在这种情况下，所记录的流一定要符合 SESF 并且在盘上一定要准备和记录 EP_map。

另外，形成数字广播流的基本流自动解码成新的基本流，并且重新多路复用成符合流格式的新的传输流，该流格式由用于标准化数字广播流的组织指定的。在这种情况下，在盘上一定要创建和记录 EP_map。

例如，假设输入流是符合 ISDB（日本数字 BS 的标准名称）的 MPEG-2 传输流，其中具有包含 HDTV 视频流和 MPEG AAC 音频流的传输流。HDTV 视频流自动解码成 SDTV 视频流，该 SDTV 视频流和原始的 AAC 音频流重新多路复用成 TS。SDTV 流和传输流两者都需要符合 ISDB 格式。

在记录介质 100 上记录数字广播流的另一个系统是进行输入传输流的透明记录，即记录不变的输入传输流，在这种情况下，EP_map 被列出和记录在盘上。

或者，输入传输流被透明地记录，即输入传输流被不变地记录，在这种情况下，TU_map 被创建和记录在盘上。

下面解释目录和文件。记录和/或再现设备 1 以下描述为 DVR（数字视频记录）。图 14 表示盘上的典型目录结构。DVR 盘的目录可以列举为：包括“DVR”目录的根目录；和包括“PLAYLIST”目录、“CLIPINF”目录、“M2TS”目录和“DATA（数据）”目录的“DVR”目录，如图 14 所示。尽管在根目录下可以创建除这些目录之外的其它目录，但这些在本实施例的应用格式中被忽略。

在“DATA”目录之下，存储有由 DVR 应用格式指定的所有文件和目录。“DVR”目录包括四个目录。在“PLAYLIST”目录下放置了实 PlayList 和虚 PlayList 的数据库文件。后面的目录可以存在于没有 PlayList 的状态中。

在“CLIPINF”下放置了 Clip 数据库。该目录也可以存在于没有 AV 流文件的状态中。在“DATA”目录中，存储有诸如数字 TV 广播的数据广播文件。

“DVR”目录存储了下述文件。即在 DVR 目录下创建的“info.dvr”以存储应用层的综合信息。在 DVR 目录下，一定有单个 info.dvr。假设该文件名称对 info.dvr 是固定的。“menu.thmb”存储了与菜单缩略图有关的信息。在 DVR 目录下，一定有 0 或 1 的标记缩略图。假设该文件名称对“menu.thmb”是固定的。如果没有菜单缩略图，该文件可以不存在。

“mark.thmb”文件存储了与标记缩略图画面有关的信息。在 DVR 目录下，一定有 0 或 1 的标记缩略图。假设该文件名称对“menu.thmb”是固定的。

如果没有菜单缩略图，该文件可以不存在。

"PLAYLIST" 目录存储了两种类型的 PlayList 文件，它们是实 PlayList 和虚 PlayList。"xxxxx.rpls"文件存储了与一个实 PlayList 有关的信息。对于每个实 PlayList 创建一个文件。文件名称是 "xxxxx.rpls"，这里 "xxxxx"表示从 0 到 9 的五个数值数字。文件扩展名一定是 "rpls"。

"yyyyy.vpls"存储了与一个虚 PlayList 有关的信息。具有文件名称 "yyyyy.vpls"的一个文件是从一个虚 PlayList 到另一个被创建的，这里 "yyyyy "表示从 0 到 9 的五个数值数字。文件扩展名一定是 "vpls"。

"CLIPINF" 目录存储了一个与每个 AV 流文件相联系的文件。"zzzzz.clps"是对应于一个 AV 流文件(Clip AV 流文件或者 Bridge-Clip 流文件)的 Clip 信息文件。文件名称是 "zzzzz.clpi"，这里 "zzzzz"表示从 0 到 9 的五个数值数字。文件扩展名一定是 "clpi "。

"M2TS" 目录存储了 AV 流文件。"zzzzz.m2ts"文件是由 DVR 系统操作的 AV 流文件。这是 Clip AV 流文件或者 Bridge-Clip AV 流文件。文件名称是 "zzzzz.m2ts"，这里 "zzzzz"表示从 0 到 9 的五个数值数字。文件扩展名一定是 "m2ts"。

"DATA"目录存储了从数据广播中传输的数据。例如该数据可以是 XML 或者 MPEG 文件。

现在解释每个目录（文件）的句法和语义。图 15 表示 "info.dvr" 文件的句法。"info.dvr"文件是由三个对象构成，即 DVRVoume()、TableOfPlayLists() 和 MakersPrivateData()。

解释图 15 所示的 info.dvr 的句法。根据来自 "info.dvr" 文件的引导字节的相对字节数，TableOfPlayList_Start_address 表示 TableOfPlayLists()的引导地址。相对字节数是从 0 开始计算的。

根据来自 "info.dvr" 文件的引导字节的相对字节数，MakersPrivateData_Start_address 表示 MakersPrivateData()的引导地址。相对字节数是从 0 开始计算的。Padding_word 与 "info.dvr" 的句法相关地插入。N1 和 N2 是可选择的正整数。每个填充字可以假设为可选择的值。

DVRVolume()存储了指示卷（盘）内容的信息。图 16 表示 DVRVoume 的句法。现在解释图 16 所示的 DVRVolume()的句法。Version_number 表示指示 DVRVolume()的版本号的四个字符字母。Version_number 编码成与 ISO646

相联系的“0045”。

长度由 32 位无符号整数表示，其表明从长度字段之后直接到 DVRVolume()尾端的字节数目。

ResumeVolume()记忆了在 Volume 中最后产生的实 PlayList 或者虚 PlayList 的文件名称。但是，当用户已经中断了实 PlayList 或者虚 PlayList 的重放时的重放位置存储在 PlayListMark()定义的恢复标记中（见图 42 和 43）。

图 17 表示 ResumeVolume()的句法。解释图 17 所示的 ResumeVolume()的句法。Valid_flag 表示当该 1 位标志分别设置为 1 或者 0 时 resume_PlayList_name 字段是有效的或是无效的。

resume_PlayList_name 的 10 字节字段表示要恢复的实 PlayList 或者虚 PlayList 的文件名称。

图 16 所示的 DVRVolume()句法中的 UIAppInfoVolume 存储了与 Volume 相关的用户接口应用程序的参数。图 18 表示 UIAppInfoVolume 的句法，现在解释其语义。Character_set 的 8 位字段表示在 Volume_name 字段中编码的字符字母的编码方法。该编码方法对应于图 19 所示的值。

Name_length 的 8 位字段表示在 Volume_name 字段中表示的 Volume 名称的字节长度。Volume_name 字段表示 Volume 的名称。从字段左边开始计数的 Name_length 数的字节数是有效字符数目，并且表示 Volume 的名称。在这些有效字符字母后面的值可以是任何值。

Volume_protect_flag 是表示卷中的内容是否能够无限制地展示给用户的标志。如果该标志设置为 1，则仅仅在用户已经成功地正确输入 PIN 号（通过口令）的情况下，卷中的内容允许被展示（再现）给用户。如果该标志设置为 0，即使在 PIN 号没有被用户输入的情况下，卷中的内容也允许展示给用户。

当用户已经将盘插入播放器时，如果该标志已经设置为 0，或者该标志设置为 1 但用户已经成功地正确输入 PIN 号的话，记录和/或再现设备 1 显示盘中的 PlayList 表。在对应 PlayList 之再现的限制是与 Volume_protect_flag 不相关的，并且是由 UIAppInfoVolume 中定义的 playback_control_flag 表示的。

PIN 是由从 0 到 9 的四个数值数字构成的，其中的每一个都是根据 ISO/IEC646 编码的。ref_thumbnail_index 字段表示加到卷中的缩略图画面的信息。如果 ref_thumbnail_index 字段是除 0xFFFF 之外的值，则缩略图画面被

加到卷中。缩略图画面存储在 menu.thumb 文件中。该画面是使用 menu.thumb 文件中的 ref_thumbnail_index 的值引用的。如果 ref_thumbnail_index 字段是 0xFFFF，则其表示缩略图画面已经被加到卷中。

解释图 15 所示的 info.dvr 句法中的 TableOfPlayList()。TableOfPlayList() 存储了 PlayList（实 PlayList 和虚 PlayList）的文件名称。记录在卷中的所有 PlayList 文件都包含在 TableOfPlayList()中，该 TableOfPlayList()表示卷中 PlayList 的缺省的重放序列。

图 20 表示 TableOfPlayList()的句法，现在解释之。TableOfPlayList()的 version_number 表示四个字符字母，其表示 TableOfPlayList 的版本号。version_number 一定要根据 ISO646 编码成“0045”。

长度是无符号的 32 位整数，其表明从长度字段之后直接到 TableOfPlayList()尾端的 TableOfPlayList()的字节数目。Number_of_PlayLists 的 16 位字段表示包括 PlayList_file_name 在内的循环（for-loop）的循环数。该数值数字一定要等于记录在卷中的 PlayList 的数目。PlayList_file_name 的 10 字节数值数字表示 PlayList 的文件名称。

图 21 表示 TableOfPlayList()句法的另一种结构。图 21 所示的句法是由图 20 所示的其中包含 UIAppInfoPlayList 的句法构成的。通过这种包括 UIAppInfoPlayList 的结构，在读出 TableOfPlayList 时简单地创建菜单画面变成有可能。下面的解释是根据使用图 20 所示的句法进行的。

解释图 15 所示的 info.dvr 中的 MakersPrivateData。提供 MakersPrivateData 是为了允许记录和/或再现设备 1 的制造者将制造者的私有数据插入 MakersPrivateData()以用于不同公司的特殊应用。每个制造者的私有数据已经被标准化为 maker_ID，以识别已经定义它的制造者。MakersPrivateData()可以包含一个或者多个 maker_ID。

如果预置制造者希望插入私有数据，并且不同制造者的私有数据已经包含在 MakersPrivateData()中，则在不擦除预先存在的旧私有数据的情况下，新私有数据加到 MakersPrivateData()。因此，在本实施例中，多个制造者的私有数据能够包含在一个 MakersPrivateData()中。

图 22 表示 MakersPrivateData 的句法。解释图 22 表示的 MakersPrivateData 的句法。TableOfPlayList()的 version_number 表示四个字符字母，其表示 TableOfPlayList 的版本号。version_number 一定要根据 ISO646 编码成“0045”。

长度是无符号 32 位整数，其表明从长度字段之后直接到 MakersPrivateData() 尾端的 TableOfPlayList()的字节数目。

Mpd_blocks_start_address 表示根据来自 MakersPrivateData()引导字节之字节数的第一个 Mpd_block()的前端地址。Number_of_maker_entries 是 16 位无代码整数，其提供 MakersPrivateData()中包括的制造者私有数据的入口号。在 MakersPrivateData()中一定不会出现具有相同 maker_ID 值的两个或多个制造者私有数据。

mpd_blocks_size 是 16 位无符号整数，其提供以 1024 字节为单位的一个 mpd_block 大小。例如，如果 Mpd_blocks_size=1，则其表示一个 Mpd_block 的大小是 1024 字节。Number_of_mpd_block 是 16 位无符号整数，其提供 MakersPrivateData()中包含的 mpd_block 数。maker_ID 是 16 位无符号整数，其表示已经创建制造者私有数据之 DVR 系统的模型数代码。编码成 maker_ID 的值是由发许可证者指定的。

maker_mode_code 是 16 位无符号整数，其表示已经创建制造者私有数据之 DVR 系统的模型数代码。编码成 maker_mode_code 的值是由制造者设置的，该制造者已经接收了格式许可。start_ mpd_block_number 是 16 位无符号整数，其表示开始制造者私有数据的 mpd_block_number 号。制造者私有数据的前端一定要与 mpd_block 的前端对齐。start_ mpd_block_number 对应于 mpd_block 循环中的变量 j。

mpd_length 是 32 位无符号整数，其表示制造者私有数据的大小。mpd_block 是其中存储了制造者的私有数据的区。MakersPrivateData()中的所有的 mpd_block 一定要是相同大小。

解释实 PlayList 文件和虚 PlayList 文件，换言之，即 xxxxx.rpls 和 yyyyy.vpls。图 23 表示 xxxxx.rpls（实 PlayList）和 yyyyy.vpls（虚 PlayList）的句法，它们具有相同的句法结构。每一个 xxxxx.rpls 和 yyyyy.vpls 都是由三个对象构成，即 PlayList()、PlayListMark()和 MakersPrivateData()。

根据来自 PlayList 文件前端的相对字节数为单位，PlayListMark_start_address 表示 PlayListMark()的引导地址。相对字节数是从 0 开始计算的。

根据来自 PlayList 文件前端的相对字节数为单位，MakersPrivateData_start_address 表示 MakersPrivateData()的引导地址。相对字

节数是从 0 开始计算的。

Padding_word（填充字）是根据 PlayList 文件的句法被插入的，其中 N1 和 N2 是可选择的正整数。每个填充字可以假设为可选择的值。

尽管其已经简要地解释了，下面仍然解释 PlayList。除 Bridge-Clip 之外的所有 Clips 中的重放域一定要由盘中的所有的 PlayList 引用。而且，两个或者多个实 PlayLists 一定不重叠相同 Clip 中由它们的 PlayItem 表示的重放域。

参照图 24A、24B 和 24C。对于所有的 Clips，存在对应的实 PlayList，如图 24A 所示。即使在编辑操作已经关闭之后也可看到该规则，如图 24B 所示。因此，所有的 Clip 一定通过引用实 PlayList 之一来看到。

参照图 24C，虚 PlayList 的重放域一定包含在重放域和 Bridge-Clip 重放域中。盘中一定不出现不由任何虚 PlayList 引用的 Bridge-Clip。

包含 PlayItem 表的实 PlayList 一定不含有 SubPlayItem。虚 PlayList 包含 PlayItem 表，并且如果包含在 PlayList()中的 CPI_type 是 EP_map 类型以及 PlayList_type 是 0（含有视频和音频的 PlayList），则虚 PlayList 可以包含一个 SubPlayItem。在本实施例的 PlayList()中，SubPlayItem 仅仅用于音频后记录。由一个虚 PlayList 拥有的 SubPlayItem 数一定是 0 或者 1。

下面解释 PlayList。图 25 表示现在要解释的 PlayList 句法。version_number 表示四个字符字母，其表示 PlayList()的版本号。version_number 根据 ISO646 编码成“0045”。长度是无符号 32 位整数，其表明从长度字段之后直接到 PlayList()尾端的 PlayList()的总字节数目。PlayList_type 是 8 位字段，其表示 PlayList 类型，图 26 示出其一个例子。

CPI_type 是一位标志，其表示由 PlayItem()和 SubPlayItem()引用的 Clip 的 CPI_type 之值。在由一个 PlayList 引用的所有 Clips 的 CPI 中定义的 CPI_type 一定具有相同值。Number_of_PlayItems 是 16 位字段，其表示出现在 PlayList 中的 PlayItem 数。

对应于预置 PlayItem()的 PlayItem_id 是由其中 PlayItem()出现在包含 PlayItem() 之循环中的序列定义的。PlayItem_id 以 0 开始。Number_of_SubPlayItems 是 16 位字段，其表示在 PlayList 中的 SubPlayItem 数。该值是 0 或者 1。附加音频流路径（音频流路径）是一种子路径类型。

解释图 25 所示的 PlayList 句法的 UIAppInfoPlayList。UIAppInfoPlayList 存储了涉及 PlayList 的用户接口应用程序的参数。图 27 表示现在要解释的

UIAppInfoPlayList 的句法。Character_set 是 8 位字段，其表示用于编码在 PlayList_name 字段中编码的字符字母的方法。该编码方法对应于与图 19 所示的表一致的值。

Name_length 是 8 位字段，其表示在 PlayList_name 字段中表示的 PlayList 名称的字节长度。PlayList_name 字段表示 PlayList 名称。从字段左边计数的 Name_length 数的字节数是有效字符数并且表示 PlayList 名称。在这些有效字符字母后面的值可以是任何值。

Record_time_and_date 是 56 位字段，其存储了记录 PlayList 的日期和时间。该字段是二进制编码的十进制（BCD）编码的年/月/日/小时/分钟/秒的 14 个数值数字。例如，2001/12/23：01：02：03 编码成 "0x20011223010203"。

持续时间（duration）是 24 位字段，其表示以小时/分钟/秒为单位的 PlayList 的总重放时间。该字段是二进制编码的十进制（BCD）编码的 6 个数值数字。例如，01：45：30 编码成 "0x014530"。

Valid_period 是 32 位字段，其表示 PlayList 的有效时间周期。该字段是 4 位二进制编码的十进制（BCD）编码的 8 个数值数字。Valid_period 用在记录和/或再现设备 1 中，即，当有效周期已经消失的 PlayList 将自动擦除时，例如，2001/05/07 编码成 "0x20010507"。

Maker_ID 是 16 位无符号整数，其表示是最近更新其 PlayList 的 DVR 播放器（记录和/或再现设备 1）的制造者。编码成 Maker_ID 的值分配给 DVD 格式的发放许可证者。Maker_code 是 16 位无符号整数，其表示是最近更新的 PlayList 的 DVR 播放器的模型数。编码成 Maker_code 的值是由制造者确定的，该制造者已经接收 DVR 格式的许可。

如果 playback_control_flag 的标志设置为 1，则其 PlayList 仅仅当用户成功地输入 PIN 号时再现。如果该标志设置为 0，则用户在不需要输入 PIN 号的情况下可以观看该 PlayList。

如果 write_protect_flag 设置为 1，则除 write_protect_flag 之外，该 PlayList 的内容既不能被擦除也不能改变。如果该标志设置为 0，用户可自由地擦除或者改变该 PlayList。如果该标志设置为 1，则在用户进行擦除、编辑或者重写 PlayList 之前记录和/或再现设备 1 显示请求用户进行重新确认的消息。

其中 write_protect_flag 设置为 0 的实 PlayList 可以存在，引用实 PlayList 之 Clip 的虚 PlayList 可以存在，并且虚 PlayList 的 write_protect_flag 可以设

置为 1。如果用户希望擦除实 PlayList，记录和/或再现设备 1 发出报警给用户以便出现前述虚 PlayList 或者在擦除实 PlayList 之前“最小化”该实 PlayList。

如果 is_played_flag 设置为 1，如图 28B 所示，则其表示自从其记录以来该 PlayList 至少再现了一次，而如果其设置为 0，则其表示自从其记录以来该 PlayList 甚至没有再现过一次。

文档（Archive）是两位字段，其表示 PlayList 是原始的还是拷贝的，如图 28C 所示。ref_thumbnail_index 的字段表示代表 PlayList 之缩略图画面的信息。如果 ref_thumbnail_index 字段是除 0xFFFF 之外的值，则代表 PlayList 之缩略图画面被加在 PlayList 中，其中 PlayList 存储在 menu.thmb 文件中。该画面使用在 menu.thmb 文件中的 Ref_thumbnail_index 之值来引用。如果 Ref_thumbnail_index 是 0xFFFF，则没有代表 PlayList 之缩略图画面被加在 PlayList 中。

下面解释 PlayItem。一个 PlayItem()基本上包含下述数据：用于指定 Clip 文件名称的 Clip_Information_file_name，成对指定 Clip 重放域的 IN_time 和 OUT_time，在 PlayList()中定义的 CPI_type 是 EP_map 类型的情况下由 IN_time 和 OUT_time 引用的 STC_sequence_id，以及表示在先 PlayItem 和当前 PlayItem 之连接条件的 Connection_Condition。

如果 PlayList 是由两个或者多个 PlayItem 构成，则在 PlayList 的全球时间轴上这些 PlayItem 成行排列，没有时间间隙或者重叠。如果在 PlayList 中定义的 CPI_type 是 EP_map 类型和当前 PlayList 不具有 BridgeSequence()，则 IN_time 和 OUT_time 对在 STC 连续域上一定表示与 STC_sequence_id 指定时间相同的时间。这种例子示于图 29。

图 30 表示这种情况，其中由 PlayList()定义 CPI_type，如果当前 PlayItem 具有 BridgeSequence()，则应用现在解释的规则。表示成 IN_time1 的在当前 PlayItem 之前的 PlayItem 之 IN_time 表示在当前 PlayItem 之 BridgeSequenceInfo()中指定的 Bridge-Clip 的时间。该 OUT_time 一定遵守随后要解释的编码限制。

表示成 IN_time2 的当前 PlayItem 的 IN_time 表示在当前 PlayItem 的 BridgeSequenceInfo()中指定的 Bridge-Clip 的时间。该 IN_time 也一定遵守后面要解释的编码限制。表示成 OUT_time2 的当前 PlayItem 的 PlayItem 之 OUT_time 表示关于由当前 PlayItem 的 STC_sequence_id 指定的 STC 连续域

的时间。

如果 PlayList()的 CPI_type 是 TU_map 类型，则成对的 PlayItem 的 IN_time 和 OUT_time 表示关于相同 Clip AV 流的时间，如图 31 所示。

PlayItem 句法示于图 32。对于图 32 所示的 PlayItem 的句法，Clip_Information_file_name 的字段表示 Clip 信息的文件名称。由该 Clip 信息文件的 ClipInfo()定义的 Clip_stream_type 一定表示 Clip AV 流。

STC_sequence_id 是 8 位字段和表示由 PlayItem 引用的连续 STC 域的 STC_sequence_id。如果在 PlayList()指定的 CPI_type 是 TU_map 类型，则该 8 位字段没有意义和设置为 0。IN_time 是 32 位字段并且用于存储 PlayItem 的重放开始时间。IN_time 的语义不同于 PlayList()中定义的 CPI_type，如图 33 所示。

OUT_time 是 32 位字段并且用于存储 PlayItem 的重放结束时间。OUT_time 的语义不同于 PlayList()中定义的 CPI_type，如图 34 所示。

Connection_condition 是 2 位字段，其表示在在先 PlayItem 和当前 PlayItem 之间的连接条件，如图 35 所示。图 36A 到 36D 表示图 35 所示的 Connection_condition 的各种状态。

参照图 37 解释 BridgeSequenceInfo。该 BridgeSequenceInfo 是当前 PlayItem 的辅助信息和包括下述信息。即，BridgeSequenceInfo 包括用于指定 Bridge_Clip AV 流文件的 Bridge_Clip_Information_file_name 和指定相应的 Clip 信息文件的 Bridge_Clip_Information_file_name（图 45）。

其也是关于由在先 PlayItem 引用的 Clip AV 流的源数据包的地址。该源数据包之后是连接 Bridge- Clip AV 流的第一源数据包。该地址叫作 RSPN_exit_from_previous_Clip。其也是关于由当前 PlayItem 引用的 Clip AV 流之源数据包的地址。该源数据包之前是连接 Bridge-Clip AV 流文件的最后的源数据包。该地址叫作 RSPN_enter_to_current_Clip。

图 37 中，RSPN_arrival_time_discontinuity 表示 Bridge-Clip AV 流的源数据包地址，其中在到达时间基上没有连续点。该地址被定义在 ClipInfo()中（图 46）。

图 38 表示 BridgeSequenceInfo 的句法。回到图 38 所示的 BridgeSequenceInfo 的句法，Bridge_Clip_Information_file_name 的字段表示对应于 Bridge_Clip_Information_file 的 Clip 信息文件的文件名称。该 Clip 信息

文件的 ClipInfo()中定义的 Clip_stream_type 一定表示'Bridge_Clip AV 流'。

RSPN_exit_from_previous_Clip 的 32 位字段是关于由在先 PlayItem 引用的 Clip AV 流的源数据包的相对地址。该源数据包之后是连接 Bridge- Clip AV 流文件的第一源数据包。RSPN_exit_from_previous_Clip 具有基于源数据包数为单位的大小，并且以 ClipInfo()中定义的 offset_SPN 值从由在先 PlayItem 引用的 Clip AV 流的第一源数据包开始计数。

RSPN_enter_to_current_Clip 的 32 位字段是关于由当前 PlayItem 引用的 Clip AV 流的源数据包的相对地址。该源数据包之前是连接 Bridge- Clip AV 流文件的最后源数据包。RSPN_ enter_to_current_Clip 具有基于源数据包数为单位的大小。RSPN_ enter_to_current_Clip 以 offset_SPN 值作为初始值开始计数，offset_SPN 值是由当前 PlayItem 引用的 Clip AV 流的第一源数据包的 ClipInfo()中定义的。

参照图 39 解释 SubPlayItem。仅仅在 PlayList()的 CPI_type 是 EP_map 类型时才允许使用 SubPlayItem()。在本实施例中，SubPlayItem 仅仅用于音频后记录。SubPlayItem()包括下述数据。第一，其包括用于指定 PlayList 中子路径引用的 Clip 的 Clip_Information_file_name。

它还包括用于指定 Clip 中子路径重放域的 SubPath_IN_time 和 SubPath_OUT_time。另外，其包括 sync_PlayItem_id 和用于指定在主路径时间轴上开始子路径再现的时间的 start_PTS_of_PlayItem。由子路径引用的 Clip AV 流一定不包含 STC 非连续点（系统时间基的非连续点）。用在子路径的 Clip 的音频采样的时钟锁定在主路径的音频采样的时钟。

图 40 表示 SubPlayItem 的句法。回到图 40 所示的 SubPlayItem 的句法，Clip_Information_file_name 的字段表示 Clip 信息文件的文件名称并且由 PlayList 中的子路径使用。该 ClipInfo()中定义的 Clip_stream_type 一定表示 Clip AV 流。

sync_PlayItem_id 的 8 位字段表示子路径类型。这里，仅仅设置了 '0x00'，如图 41 所示，同时其它值被保留为今后使用。

sync_PlayItem_id 的 8 位字段表示 PlayItem 的 PlayItem_id，PlayItem 包含在主路径时间轴上的子路径的重放开始时间。对应于预置 PlayItem 的 PlayItem_id 的值在 PlayList()中定义（图 25）。

sync_start_PTS_of_PlayItem 的 32 位字段表示在主路径时间轴上的子路

径的重放开始时间，并且表示由 sync_PlayItem_id 引用的 PlayItem 上的 PTS（显示时间戳）的高 32 位。SubPath_IN_time 的高 32 位字段存储了子路径的重放开始时间。SubPath_IN_time 表示对应于子路径中第一显示单元的 33 位 PTS 的高 32 位。

SubPath_OUT_time 的高 32 位字段存储了子路径的重放结束时间。SubPath_OUT_time 表示由下述方程计算的 Presentation_end_TS 之值的高 32 位。

Presentation_end_TS=PTS_OUT+AU_duration。

这里，PTS_OUT 是对应于 SubPath 最后显示单元的 33 位长度的 PTS，AU_duration 是基于 SubPath 最后显示单元的显示周期 90kHz。

下面，解释图 23 所示的 xxxxx.rpls 和 yyyyy.vpls 句法中的 PlayListMark()。与 PlayList 有关的标记信息被存储该 PlayListMark 中。图 42 表示 PlayListMark 的句法。回到图 42 所示的 PlayListMark 的句法，version_number 是四个字符字母，其表示该 PlayListMark()的版本号。version_number 一定要根据 ISO646 编码成“0045”。

长度（Length）是无符号 32 位整数，其表示从长度字段之后直接到 PlayListMark()尾端的 PlayListMark()字节数目。Number_of_PlayListMarks 是表示 16 位无符号整数，其表示存储在 PlayListMark 中的标记数目。Number_of_PlayListMarks 可以是 0。Mark_type 是标记类型的 8 位字段并且被编码在图 43 表示的表中。

Mark_time_stamp 的 32 位文件存储表示由标记指定的点的时间戳。Mark_time_stamp 的语义不同于在 PlayList()中定义的 CPI_type，如图 44 所示。PlayItem_id 是指定 PlayItem 的 8 位字段，其中输入标记。对应于预置 PlayItem 的 PlayItem_id 的值被定义在 PlayList()中（见图 25）。

Character_set 的 8 位字段表示在 mark_name 字段中编码的字符字母的编码方法。编码方法对应于图 19 所示的值。Name_length 的 8 位字段表示在 mark_name 字段所示的标记名称的字节长度。mark_name 字段表示在 mark_name 字段中表明的标记名称。对应于 Name_length 数的从该字段左边开始的字节数是有效字符字母和表示该标记名称。在 mark_name 字段中，在这些有效字符字母后面的值可以是任意的。

ref_thumbnail_index 的字段表示加到标记的缩略图画面的信息。如果

ref_thumbnail_index 的字段不是 0xFFFF，则缩略图画面被加到其标记，同时缩略图画面存储在 mark.thmb 文件中。该画面在 mark.thmb 文件中被引用，其使用 ref_thumbnail_index 的值，如后面的解释。如果 ref_thumbnail_index 的字段是 0xFFFF，则其表示没有缩略图画面被加到标记。

现在解释 Clip 信息文件。zzzzz.clpi(Clip 信息文件)是由六个对象构成的，如图 45 所示。这些是 ClipInfo()、STC_Info()、Program()、CPI()、ClipMark()和 MarkersPrivateData()。对于 AV 流(Clip AV 流或者 Bridge-Clip AV 流)和对应的 Clip 信息文件,使用相同 的“zzzzz” 数字串。

解释回到图 45 所示的 zzzzz.clpi(Clip 信息文件)的句法。ClipInfo_Start_address 表示 ClipInfo()的前端地址,其中具有从 zzzzz.clpi 文件的前端字节作为单位的相对字节数。该相对字节数是从 0 计数。

STC_Info_Start_address 表示 STC_Info 的前端地址,其中具有从 zzzzz.clpi 文件的前端字节作为单位的相对字节数。ProgramInfo_Start_address 表示 ProgramInfo()的前端地址,其中具有从 zzzzz.clpi 文件之前端字节作为单位的相对字节数。该相对字节数是从 0 计数。CPI_Start_address 表示 CPI()的前端地址,其中具有从 zzzzz.clpi 文件之前端字节作为单位的相对字节数。该相对字节数是从 0 计数。

ClipMark_Start_address 表示 ClipMark()的前端地址,其中具有从 zzzzz.clpi 文件之前端字节作为单位的相对字节数。该相对字节数是从 0 计数。MakersPrivateData_Start_address 表示 MakersPrivateData()的前端地址,其中具有从 zzzzz.clpi 文件之前端字节作为单位的相对字节数。该相对字节数是从 0 计数。Padding_word 根据 zzzzz.clpi 文件的句法被插入。N1，N2，N3，N4 和 N5 一定是 0 或者是可选择的正整数。对应的填充字也可以假设为可选择值。

现在解释 ClipInfo。图 46 表示 ClipInfo 的句法。在 ClipInfo()中存储了对应的 AV 流文件的属性信息（Clip AV 流或者 Bridge-Clip AV 流文件）。

回到图 46 所示的 ClipInfo 的句法，version_number 是四个字符字母，其表示该 ClipInfo()的版本号。version_number 一定要根据 ISO646 编码成“0045”。长度是无符号 32 位整数，其表明从长度字段之后直接到 ClipInfo()尾端的 ClipInfo()的字节数目。Clip_stream_type 的 8 位字段表示对应于 Clip 信息文件的 AV 流的类型，如图 47 所示。下面解释对应 AV 流的流类型。

Offset_SPN 的 32 位字段给出 AV 流(Clip AV 流或者 Bridge-Clip AV 流)第一源数据包的第一源数据包数的源数据包数的偏移值。当 AV 流文件首先记录在盘上时，该 Offset_SPN 一定为 0。

参照图 48，当 AV 流文件的开始部分是通过编辑被擦除时，该 Offset_SPN 可以假定为除 0 之外的值。在本实施例中，引用 Offset_SPN 的相对源数据包数（相对地址）经常以 RSPNxxx 形式说明，这里 xxx 经更新使得 RSPN_xxx 是 RAPN_EP_start。相对源数据包数是以源数据包数作为单位定大小的并且是从 AV 流文件第一源数据包数开始计数，其中具有 Offset_SPN 值作为初始值。

从 AV 流文件第一源数据包到引用相对源数据包数（SPN_xxx）的源数据包的源数据包数是通过下述方程计算的:

SPN_xxx= RSPN_xxx- Offset_SPN

图 48 表示其中 Offset_SPN 为 4 的例子。

TS_recording_rate 是 24 位无符号整数，其为 AV 流所要求的给 DVR 驱动（写单元 22）或者来自 DVR 驱动（读出单元 28）的输入/输出位速率。Record_time_and_date 是 56 位字段，用于存储对应于 Clip 之 AV 流的记录日期和时间，并且以 4 位二进制编码的十进制（BCD）进行编码表示成年/月/日/小时/分钟的 14 个数值数字。例如，2001/2/23: 01: 02: 03 编码成“0x20011223010203”。

持续时间（duration）是 24 位字段，其表示基于到达时钟的以小时/分钟/秒的 Clip 总重放时间。该字段是 4 位二进制编码的十进制（BCD）编码的 6 个数值数字。例如，01: 45: 30 编码成“0x014530”。

标志 time_controlled_flag 表示 AV 流文件的记录模式。如果该 time_controlled_flag 是 1，则表示记录模式是这种模式，其中文件大小是正比于自记录以来过去的时间，结果由下述方程所示的条件:

Ts_average_rate*192/188*(t-start_time)-α<=size_clip(t)

<=TS_average_rate*192/188*(t-start_time)+ α

这里 TS_average_rate 是用字节/秒表示的 AV 流文件传输流的平均位速率。

上述方程中，t 表示以秒为单位的时间，start_time 是当 AV 流文件的第一源数据包被记录时的时间点。size_clip(t)是 10*192 字节，α 是依赖于

TS_average_rate 的常数。

如果 time_controlled_flag 设置为 0，则其表示记录模式没有受到控制，使得记录所过去的时间正比于 AV 流的文件大小。例如，输入传输流以透明方式进行记录。

如果 time_controlled_flag 设置为 1，TS_average_rate 的 24 位字段表示用在上述方程中的 TS_average_rate 的值。如果 time_controlled_flag 设置为 0，则该字段没有意义和一定要设置为 0。例如，可变位速率传输流是通过下述顺序编码的：首先，传输速率被设置成 TS_recording_rate。视频流用可变位速率编码。传输数据包通过不采用空数据包被间断地编码。

RSPN_arrival_time_discontinuity 的 32 位字段是位置的相对地址，在该位置，到达时间基不连续性是在 Bridge-Clip AV 流文件上再现的。RSPN_arrival_time_discontinuity 是以源数据包数作为单位定大小的，并且从在 ClipInfo()中定义的作为来自 Bridge-Clip AV 流文件之第一源数据包的 offset_SPN 的值计数。Bridge-Clip AV 流文件中的绝对地址是基于上述方程计算的：

SPN_xxx=RSPN_xxx-offset_SPN.

Reserver_for_system_use 的 144 位字段是保留给系统的。如果 is_format_identifier_valid 标志是 1，则其表示 format_identifier 的字段是有效的。如果 is_format_identifier_valid 标志是 1，则其表示 format_identifier 字段是有效的。如果 is_original_network_ID_valid 标志是 1，则其表示 is_original_network_ID_valid 的字段是有效的。如果标志 is_transport_stream_ID_valid 是 1，则其表示 transport_stream_ID 字节是有效的。如果 is_servece_ID_valid 标志是 1，则其表示 servece_ID 字段是有效的。

如果 is_country_code_valid 标志是 1，则其表示字段 country_code 是有效的。Format_identifier 的 32 位字段表示由传输流中的注册说明符（ISO/IEC13818-1 中定义的）拥有的 format_identifier 的值。original_network_ID_的 16 位字段表示在传输流中定义的 original_network_ID 的值。

Servece_ID 中的 16 位字段表示在传输流中定义的 Servece_ID 的值。country_code 的 24 位字段表示由 ISO3166 定义的国家代码。每个字符代码被 ISO8859-1 编码。例如，日本表示成“JPN”和编码成“0x4A0x500x4E”。

stream_format_name 是 ISO-646 的 15 个字符代码,其表示提供传输流流定义之格式组织的名称。该字段中的无效字节具有 "0xFF" 值。

format_identifier 、 original_network_ID 、 transport_stream_ID 、 servece_ID,country_code 和 stream_format_name 表示传输流的服务提供者。这允许识别对音频或者视频流的编码限制和除音视流或者 SI（服务信息）之外的私有数据的流定义。这些信息能够被用来检查解码器是否能够解码该流。如果这种解码是可能的，则该信息可以在开始解码之前用来初始化解码器系统。

现在解释 STC_Info。不包含 STC 不连续点（系统时间基的不连续点）的 MPEG-2 传输流中的时间域叫作 STC_sequence。在 Clip 中，STC_sequence 由 STC_sequence_id 的值指定。图 50A 和 50B 表示连续 STC 域。相同的 STC 值决不出现在相同 STC_sequence 中，尽管 Clip 的最大时间长度是有限的，如后面的解释。因此，相同 PTS 值也决不出现在相同 STC_sequence 中。如果 AV 流包含 N 个 STC 不连续点，这里 N>0，则 Clip 系统时间基被分开成（N+1）个 STC_sequence。

STC_Info 存储了地点的地址，其中产生 STC 了不连续性（系统时间基不连续性）。正如参照图 51 的说明，RSPN_STC_start 表示地址，且在由第 k+1 个 RSPN_STC_start 引用的源数据包之到达时间点的开始和在最后源数据包之到达时间点的结束。

图 52 表示 STC_Info 的句法。回到图 52 所示的 STC_Info 的句法，version_number 是四个字符字母，其表示该 STC_Info() 的版本号。version_number 一定要根据 ISO646 编码成 "0045"。

长度是无符号 32 位整数，其表明从该长度字段正后面到 STC_Info 尾端的 STC_Info()字节数目。如果 CPI()的 CPI_type 表示 TU_map 类型，则 0 可以设置在该长度字段中。如果 CPI()的 CPI_type 表示 EP_map 类型，则 num_of_STC_sequence 一定是不小于 1 的值。

num_of_STC_sequence 的 8 位无符号整数表示 Clip 中的序列数。该值表示在该字段后面的循环数。对应于预置 STC_sequence 的 STC_sequence_id 是由次序(order)定义的，该次序中出现对应于包含 RSPN_STC_start 之循环中的 STC_sequence 的 RSPN_STC_start。STC_sequence_id 由 0 开始。

RSPN_STC_start 的 32 位字段表示在此 STC_sequence 在 AV 流文件开始

的地址。RSPN_STC_start 表示在 AV 流文件中产生系统时间基不连续性的地址。RSPN_STC_start 也可以是源数据包的相对地址，源数据包具有 AV 流中新系统时间基的第一个 PCR。RSPN_STC_start 是基于源数据包数的大小，并且是从具有在 ClipInfo()中定义的作为初始值之 offset_SPN 的 AV 流文件第一源数据包计数。在该 AV 流文件中，绝对地址是通过上述方程计算的，即：

SPN_xxx=RSPN_xxx-offset_SPN.

现在参照图 53 解释图 45 所示的 zzzzz.clip 句法中的 ProgramInfo。具有 Clip 中下述特征的时间域叫作 program_sequence。这些特征是：PCR_PID 的值不改变，音频基本流数也不改变，对应视频流中的 PID 值不改变，由其 VideoCodingInfo 定义的编码信息不改变，音频基本流数也不改变，对应音频流中的 PID 值不改变，以及由其 AudioCodingInfo 定义的编码信息不改变。

Program_sequence 在相同时间点上仅仅具有一个系统时间基。Program_sequence 在相同时间点上具有单个 PMT。ProgramInfo()存储 Program_sequence 开始的地点地址。RSPN_program_sequence_start 表示地址。

图 54 表示 ProgramInfo 的句法。回到图 54 所示的 ProgramInfo，version_number 是四个字符字母，其表示该 ProgramInfo()的版本号。version_number 一定要根据 ISO646 编码成“0045”。

长度是无符号 32 位整数，其表明从该长度字段正后面到 ProgramInfo 尾端的 ProgramInfo()字节数目。如果 CPI()的 CPI_type 表示 TU_map 类型，则该长度字段可以设置为 0。如果 CPI()的 CPI_type 表示 EP_map 类型，则 number_of_program 一定是不小于 1 的值。

number_of_program_sequence 的 8 位无符号整数表示 Clip 中的 program_sequence 数。该值表示在该字段后面的循环数。如果在 Clip 中 program_sequence 不改变，则 1 一定被设置在 program_sequence 数中。RSPN_program_sequence_start 的 32 位字段是相对地址，其中在 AV 流上开始该节目序列。

RSPN_program_sequence_start 是以源数据包数作为单位定大小的，并且是从 AV 流文件的第一源数据包开始以 ClipInfo()中定义的 offset_SPN 的值计数。在该 AV 流文件中，绝对地址是通过下式计算的，即：

SPN_xxx=RSPN_xxx-offset_SPN.

在循环句法中的 RSPN_program_sequence_start 的值一定要以升序出现。

PCR_PID 的 16 位字段表示传输数据包的 PID，传输数据包包含对 program_sequence 有效的有效 PCR 字段。Number_of_audios 的 8 位字段表示包含 audio_stream_PID 和 AudioCodingInfo()之循环数。Video_stream_PID 的 16 位字段表示传输数据包的 PID，传输数据包包含对 program_sequence 有效的视频流。在该字段后面的 VideoCodingInfo()一定解释由其 Video_stream_PID 引用的视频流的内容。

Audio_stream_PID 的 16 位字段表示传输数据包的 PID，传输数据包包含对其 program_sequence 有效的音频流。在该字段后面的 AudioCodingInfo()一定解释由其 audio_stream_PID 引用的视频流的内容。

其中 Video_stream_PID 的值出现在循环句法中的次序一定等于对 program_sequence 有效的 PMT 中视频流 PID 编码的序列。另外，其中 audio_stream_PID 的值出现在循环句法中的次序一定等于对 program_sequence 有效的在 PMT 中对音频流 PID 编码的序列。

图 55 表示图 54 所示的 ProgramInfo 句法中的 VideoCodingInfo 的句法。回到图 55 所示的 VideoCodingInfo 的句法，video_format 的 8 位字段表示对应于 ProgramInfo()的 Video_stream_PID 的视频格式，如图 56 所示。

参照图 57，frame_rate 的 8 位字段表示对应于 ProgramInfo()中的 video_stream_PID 的视频帧速率。Display_aspect_ratio 的 8 位字段表示对应于 ProgramInfo()的 video_stream_PID 的视频显示纵横比。

图 59 表示图 54 所示的 ProgramInfo 句法中的 AudioCodingInfo 的句法。回到图 59 所示的 AudioCodingInfo 的句法，audio_format 的 8 位字段表示对应于 ProgramInfo()的 audio_stream_PID 之音频编码方法，如图 60 所示。

audio_component_type 的 8 位字段表示对应于 ProgramInfo()的 audio_stream_PID 之音频元件类型,如图 61 所示,同时 sampling_frequency 的 8 位字段表示对应于 ProgramInfo()的 audio_stream_PID 之音频采样频率,如图 62 所示。

解释图 45 所示 zzzzz.clip 句法中的 CPI（特征点信息）。CPI 用于将 AV 流中的时间信息与其文件中的地址相关联。CPI 有两种类型，即 EP_map 和 TU_map。图 63 中，如果 CPI()中的 CPI_type 是 EP_map，则其 CPI()包含 EP_map。图 64 中，如果 CPI()中的 CPI_type 是 TU_map，则其 CPI()包含 TU_map。一个 AV 流具有一个 EP_map 或者一个 TU_map。如果 AV 流是 SESF

传输流，则对应的 Clip 一定拥有 EP_map。

图 65 表示 CPI 的句法。回到图 65 所示的 CPI 的句法，version_number 是四个字符字母，其表示该 CPI()的版本号。version_number 一定要根据 ISO646 编码成“0045”。长度是无符号 32 位整数，其表明从该长度字段正后面到 CPI() 尾端的字节数目。CPI_type 是 1 位标志并且表示 Clip 的 CPI 类型，如图 66 所示。

解释图 65 所示 CPI 句法中的 EP_map。有两种类型 EP_map，即用于视频流的 EP_map 和用于音频流的 EP_map。EP_map 中的 EP_map_type 区分开这些 EP_map 类型。如果 Clip 包含一个或者多个视频流，则一定使用用于视频流的 EP_map。如果 Clip 不包含视频流但包含一个或多个音频流，则一定使用用于音频流的 EP_map。

参照图 67 解释用于视频流的 EP_map。用于视频流的 EP_map 具有数据 stream_PID，PTS_EP_start 和 RSPN_EP_start。stream_PID 表示传输视频流的传输数据包的 PID。PTS_EP_start 表示从视频流序列标头（header）开始的存取单元的 PTS。RSPN_EP_start 表示源数据包的地址，该源数据包包括由 AV 流中的 PTS_EP_start 引用的存取单元的第一字节。

叫做 EP_map_for_one_stream_PID()的子表是从一个视频流创建的，该视频流由具有相互相同 PID 的传输数据包传输。如果多个视频流存在于 Clip 中，则 EP_map 可以包含多个 EP_map_for_one_stream_PID()。

用于音频流的 EP_map 具有数据 stream_PID、PTS_EP_start 和 RSPN_EP_start。stream_PID 表示传输音频流的传输数据包的 PID。PTS_EP_start 表示在音频流中的存取单元的 PTS。RSPN_EP_start 表示源数据包的地址，该源数据包包括由 AV 流的 PTS_EP_start 引用的存取单元的第一字节。

叫做 EP_map_for_one_stream_PID()的子表是从个音频流创建的，该音频流由具有相互相同 PID 的传输数据包传输。如果多个音频流存在于 Clip 中，则 EP_map 可以包含多个 EP_map_for_one_stream_PID()。

回到在 EP_map 和 STC_Info 之间的关系，EP_map_for_one_stream_PID() 是在与 STC 中的不连续点无关的表中创建。将 RSPN_EP_start 的值与 STC_Info()中定义的 RSPN_STC_start 的值进行比较，则显示出属于对应 STC_sequence 的 EP_map 的数据边界（见图 68）。EP_map 一定具有对由相同

PID 传输的连续流范围的一个 EP_map_for_one_stream_PID。在图 69 所示的情况下，节目#1 和节目#2 具有相同的视频 PID，但是，数据范围不是连续的，使得必须对每个程序一定提供 EP_map_for_one_stream_PID。

图 70 表示 EP_map 句法。通过对图 70 所示 EP_map 句法的解释，EP_type 是 4 位字段并且显示 EP_map 入口点类型，如图 71 所示。EP_type 表示在该字段后面的数据字段的语义。如果 Clip 包括一个或多个视频流，则 EP_type 一定被设置成 0（'视频'）。另外，如果 Clip 不包括视频流但包含一个或多个音频流，则 EP_type 一定被设置成 1（'音频'）。

Number_of_stream_PID 的 16 位字段表示具有在 EP_map 中 Number_of_stream_PID 作为变量的循环的循环的次数。Stream_PID(k)的 16 位字节表示传输数据包的 PID，该传输数据包传输由 EP_map_for_one_stream_PID(num_EP_entries(k))引用的数 k 基本流(视频或者音频流)。如果 EP_type 是 0（'视频'），其基本流一定是视频流。如果 EP_type 等于 1（'音频'），其基本流一定是音频流。

num_EP_entries(k)的 16 位字段表示由 EP_map_entries(k)引用的 num_EP_entries(k)。EP_map_for_one_stream_PID_Start_ address(k)：该 32 位字段表示相对地址位置，在此，EP_map_for_one_stream_PID (num_EP_entries(k))开始于 EP_map()。该值是由从 EP_map()第一字节开始的大小表示的。

Padding_word 一定根据 EP_map()句法插入。X 和 Y 一定是可选择的正整数。对应填充字可以假设任意可选值。

图 72 表示 EP_map_for_one_stream_PID 的句法。通过对图 72 所示 EP_map_for_one_stream_PID 句法的解释，PTS_EP_start 的 32 位字段的语义不同于 EP_map()定义的 EP_type。如果 EP_type 等于 0（'视频'），则该字段具有开始于视频流序列标头的存取单元的 33 位精密 PTS 的高 32 位。如果 EP_type 等于 1（'音频'），该该字段具有音频流存取单元的 33 位精密 PTS 的高 32 位。

RSPN_EP_start 的 32 位字段的语义不同于 EP_map()定义的 EP_type。如果 EP_type 等于 0（'视频'），则该字段表示源数据包的相对地址，源数据包包括由 AV 流中 PTS_EP_start 引用的存取单元序列标头的第一字节。另外，如果 EP_type 等于 1（'音频'），则该字段表示源数据包的相对地址，源数据

包包括由 AV 流中 PTS_EP_start 引用的存取单元音频流中的第一字节。

RSPN_EP_start 是基于源数据包数作为单位定大小的，并且是从 AV 流文件的第一源数据包开始以 ClipInfo()中定义的 offset_SPN 的值为初始值计数。在该 AV 流文件中，绝对地址是通过下式计算的，即：

SPN_xxx=RSPN_xxx-offset_SPN.

注意，句法中的 RSPN_EP_start 的值一定以升序出现。

现在参照图 73 解释 TU_map。TU_map 形成基于源数据包到达时钟（到达时间基的时计）的时间轴。该时间轴叫做 TU_map_time_axis。TU_map_time_axis 的原点由 TU_map() 中的 offset_time 表示。TU_map_time_axis 从 offset_time 以预置单位被分割，该单位叫做 time_unit。

在 AV 流的每个 time_unit 中，第一完全形式之源数据包 AV 流文件上的地址存储在 TU_map 中。这些地址叫做 RSPN_time_unit_start。在 TU_map_time_axis 上开始第 k 个(k□0)time_unit 的时间叫做 TU_start_time(k)。该值是基于下述方程计算的：

TU_start_time(k)= offset_time+k* time_unit_size.

注意，TU_start_time(k)具有 45kHz 的精确度。

图 74 表示 TU_map 的句法。通过解释图 74 所示的 TU_map 的句法，offset_time 的 32 位字段给出与 TU_map_time_axis 相关的偏移时间。该值表示与 Clip 中第一 time_unit 相关的偏移时间。offset_time 的大小是以从 27MHz 精确到达时钟作为单位得出的 45kHz 为基础的。如果 AV 流将作为新 Clip 进行记录，则 offset_time 一定设置为 0。

time_unit_size 的 32 位字段提供 time_unit 的大小，并且是以从 27MHz 精确到达时钟作为单位得出的 45kHz 为基础的。最好是，time_unit_size 不长于 1 秒（time_unit_size□45000）。number_of_time_unit_entries 的 32 位字段表示存储在 TU_map()中的入口数。

RSN_time_unit_start 的 32 位字段表示 AV 流中每个 time_unit 开始的位置的相对地址。RSN_time_unit_start 是基于源数据包数作为单位的大小，并且用从 AV 流文件第一源数据包开始的在 ClipInfo()中定义的 offset_SPN 的值作为初始值计数。AV 流文件中的绝对地址是通过下式计算的：

SPN_xxx=RSPN_xxx-offset_SPN.

注意，句法循环中的 RSN_time_unit_start 的值一定要以升序出现。如果

在数(k+1)的 time_unit 中没有源数据包，则数(k+1)的 RSN_time_unit_start 一定等于数 k 的 RSN_time_unit_start。

通过解释图 45 所示的 zzzzz.clip 句法中的 ClipMark，ClipMark 是与 Clip 有关的标记信息，并且存储在 ClipMark 中。该标记不是通过用户设置的，但是是通过记录器(记录和/或再现设备 1)设置的。

图 75 表示 ClipMark 的句法。通过解释图 75 所示的 ClipMark 的句法，version_number 是四个字符字母，其表示该 ClipMark 的版本号。version_number 一定要根据 ISO646 编码成“0045”。

长度是无符号 32 位整数，其表明从长度字段之后直接到 ClipMark()尾端的 ClipMark()的字节数目。Number_of_Clip_marks 是 16 位无符号整数，其表示存储在 ClipMark 中的标记数并且可以等于 0。Mark_type 是 8 位字段，其表示标记类型和根据图 76 所示的表被编码。

Mark_time_stamp 是 32 位字段和存储了表示具有专用标记的指针的时间戳。Mark_time_stamp 的语义不同于 PlayList()中的 CPI_type，如图 77 所示。

如果 CPI()中的 CPI_type 表示 EP_map 类型，则该 8 位字段表示放置有 Mark_time_stamp 的连续 STC 域的 STC_sequence_id。如果 CPI()中的 CPI_type 表示 TU_map 类型，则该 8 位字段没有意义但设置为 0。Character_set 的 8 位字段表示在 mark_name 字段中编码的字符字母的表示方法。该编码方法对应于图 19 所示值。

Name_length 的 8 位字段表示在 mark_name 字段中所示的标记名称的字节长度。该 mark_name 字段表示标记名称。对应于 Name_length 数的从该字段左边开始的字节数是有效字符数和表示标记名称。在 mark_name 字段中，在这些有效字符字母之后的值可以是任意的。

ref_thumbnail_index 的字段表示附加到标记的缩略图画面的信息。如果 ref_thumbnail_index 字段是不同于 0xFFFF 的值，该缩略图画面被附加到其标记，缩略图画面存储在 mark.thumb 文件中。该画面是使用 mark.thumb 文件中的 ref_thumbnail_index 之值引用的。如果 ref_thumbnail_index 字段是等于 0xFFFF 的值，则缩略图画面不附加到其标记。

MakerPrivateData(制造者个人数据)已经参照图 22 进行了说明，因此这里不再特别说明。

接下来说明 thumbnail_information(缩略图信息)。极小画面存储于

menu.thmb（菜单）文件或 mark.thmb（标记）文件中。这些文件具有相同的句法结构并拥有唯一的 Thumbnail()。menu.thmb 文件存储表示各自 PlatyLists 的画面。菜单缩略图的整体存储于唯一的 menu.thmb 文件中。

mark.thmb 文件存储标记缩略图画面，它是表示标记点的画面。对应于 PlatyLists 和 Clips 的整体的标记缩略图的整体存储于唯一的 mark.thmb 文件中。由于频繁地增加或删除缩略图，增加和部分删除的操作必须可以容易并迅速地得到执行。为此，Thumbnail()具有块结构。画面数据被分为多个部分，每个部分存储于一个 tn_block（tn 块）中。一个画面数据被存储于连续的 tn_block 中。在 tn_block 的串中，可能存在未使用的 tn_block 块。唯一缩略图画面的字节长度是可变的。

图 78 显示了 menu.thmb 和 mark.thmb 的句法，并且图 79 显示了图 78 所示的 menu.thmb 和 mark.thmb 的句法结构中的 Thumbnail（缩略图）的句法。如图 79 所示的 Thumbnail（缩略图）的句法的说明，version_number（版本号）是表示这个 Thumbnail()的版本号的四个字符的字母。version_number 根据 ISO 646 必须被编码为 “0045”。

长度为 32 位的无符号整数，用于指示从长度字段的尾部到 Thumbnail()的末端的 MakerPrivateData()的字节数。以到 Thumbnail()的前端字节的字节相对数作为单位的话，tu_block_strat_address（tu 块开始地址）是指示第一个 tn_block 的前端字节地址的 32 位无符号整数。从 0 计数相对字节数。Number_of_thumbnails 是 16 位无符号整数，它给出了包含于 Thumbnail()中的缩略图画面的条目数。

tu_block_size 是 16 位无符号整数，以 1024 个字节作为单位给出一个 tn_block 的大小。例如，如果 tn_block_size=1，它表示一个 tn_block 的大小是 1024 字节。number_of_tn_blocks 是 116 位的无符号整数，它表示这个 Thumbnail()中的 tn_block 的条目数。thumbnail_index 是 16 位无符号整数，它表示从 thumbnail_index 字段开始的一个循环的由缩略图信息代表的缩略图画面的索引号。值 0xFFFF 不必用作 thumbnail_index。这个 thumbnail_index 由 UIAppInfoVolume()、UIAppInfoPlayList()、PlayListMark()和 ClipMark()中的 ref_thumbnail_index 参照。

Thumbnail_picture_format 是 8 位无符号整数，代表缩略图画面的格式，并假设图 80 所示的值。在表中，DCF 和 PNG 仅在 menu.thumb 中被允许。

标记缩略图必须假设“0x00”值（MPEG-2 视频 I 画面）。

Picture_data_size 是 32 位无符号整数，以字节为单位表示缩略图画面的字节长度。Start_tn_block_number 是 16 位无符号整数，表示缩略图画面数据开始的 tn_block 的 tn_block 序号。缩略图画面数据的前端必须与 tn_block 的前端相一致。tn_block 序号开始于 0 并相关于 tn_block 的循环中的变量 k 的值。

X_picture_length 是 16 位无符号整数，表示缩略图画面的帧的水平方向的像素数。Y_picture_length 是 16 位无符号整数，表示缩略图画面的帧的垂直方向的像素数。tn_block 是其中存储缩略图画面的区域。Thumbnail()中的所有 tn_block 大小相同（固定长度）并由 tn_block_size 定义大小。

图 81A 和 81B 概要地显示了缩略图画面数据是如何存储在 tn_block 中的。如图 81A 和 81B 所示，如果缩略图画面在 tn_block 的前端开始，并且大小超过 1 tn_block，则它用下一个接着的 tn_block 存储。通过这样做，具有可变长度的数据可以作为固定长度的数据管理，使得可以用更简单的处理进行删除的编辑。

现在说明 AV 流文件。AV 流文件存储于“M2TS”目录（图 14）中。有两种类型的 AV 流文件，即 Clip(剪辑)AV 流文件和 Bridge-Clip(桥式剪辑)AV 流文件。两种 AV 流文件必须是如后面所定义的 DVR MPEG-2 传输流文件的结构。

首先说明 DVR MPEG-2 传输流。DVR MPEG-2 传输流的结构示于图 82。AV 流文件具有 DVR MPEG-2 传输流的结构。DVR MPEG-2 传输流由整数个联合单元（Aligned unit）组成。联合单元的大小是 6144 个字节（2048*3 字节）。联合单元从源数据包的第一个字节开始。源数据包 192 个字节长。一个源数据包包括 TP_extra_header 和传输数据包。TP_extra_header 长度为 4 个字节，传输数据包 188 个字节长。

一个联合单元由 32 个源数据包组成。DVR MPEG-2 传输流中的最后的联合单元也由 32 个源数据包组成。因此，DVR MPEG-2 传输流在联合单元的边界结束。如果记录在盘上的输入传输流的传输数据包的个数不是 32 的倍数，则具有空数据包（PID 的传输数据包=0x1FFFF）的源数据包必须用作最后的联合单元。文件系统不必使用 DVR MPEG-2 传输流中的额外信息。

图 83 显示了 DVR MPEG-2 传输流的记录器模型。图 83 中所示的记录器

是用于规定记录过程的概念性模型。DVR MPEG-2 传输流遵从这个模型。

现在说明 MPEG-2 传输流的输入时序（timing）。输入的 MPEG-2 传输流是全部传输流或部分传输流。输入的 MPEG-2 传输流必须遵从 ISO/IEC13818-1 或 ISO/IEC 13818-9。MPEG-2 传输流的第 i 个字节在时间 t(i) 同时输入给 T-STD（ISO/IEC 13818-1 规定的传输流系统目标解码器）和源分包器。Rpk 是传输数据包的输入速率的瞬时最大值。

27MHz PLL 52 产生 27MHz 的时钟频率。27MHz 的时钟频率被锁定于 MPEG-2 传输流的程序时钟基准（PCR）的值。到达时间时钟计数器 53 计数 27MHz 频率的脉冲。Arrival_time_clock(i)是到达时间时钟计数器在时间 t(i) 的计数值。

源分包器 54 将 TP_ectra_header 附加到传输数据包的整体上以创建源数据包。Arrival_time_stamp(到达时间戳)表示传输数据包的第一个字节到达 T-STD 和源分包器两者的时间。Arrival_time_stamp(k)是如通过下列方程代表的 Arrival_time_clock(k)的采样值:

Arrival_time_stamp(k)= Arrival_time_clock(k)%230

这里 k 表示传输数据包的第一个字节。

如果两个相邻传输数据包之间的时间间隔是 230/27000000 秒（大约 40 秒）或更长，两个传输数据包的 Arrival_time_stamp 的差值应该设置为 230/27000000 秒。为这种情况提供记录器。

平滑缓冲器 55 使输入传输流的位速率平滑。平滑缓冲器不必溢出。Rmax 是当平滑缓冲器不为空时源数据包从平滑缓冲器的输出位速率。如果平滑缓冲器为空，平滑缓冲器的输出位速率为 0。

接下来，说明 DVR MPEG-2 传输流的记录器模型的参数。Rmax 的值由在与 AV 流文件相关的 ClipInfo()中定义的 TS_recording_rate（TS 记录速率）给出。这个值可以从下列方程进行计算:

Rmax= TS_recording_rate*192/188

这里 TS_recording_rate 的值是字节大小/秒钟。

如果输入传输流是 SESF 传输流，则 Rpk 必须等于在与 AV 流文件相关的 ClipInfo()中定义的 TS_recording_rate。如果输入传输流不是 SESF 传输流，则基准可以是例如在 MPEG 2 传输流的描述符中定义的值，如用于这个值的 maximum_bitrate_descriptor（最大位速率描述符）或 partial_stream_descriptor

（部分流描述符）。

如果输入传输流是 SESF 传输流，则平滑缓冲器大小为 0。如果输入传输流不是 SESF 传输流，则基准可以是例如在 MPEG 2 传输流的描述符中定义的值，例如在 smoothing_buffer_descriptor（平滑缓冲器描述符）中、在 short_smoothing_buffer_descriptor（短平滑缓冲器描述符）中、或在 partial_transport_stream_descriptor（部分传输流描述符）中定义的值。

对于记录器和播放器（再现设备），需要提供足够大小的缓冲器。缺省的缓冲器大小是 1536 字节。

接下来描述 DVR MPEG 2 传输流的播放器模型。图 84 显示了 DVR MPEG 2 传输流的播放器模型。这是规定再现过程的概念性模型。DVR MPEG 2 传输流遵从这个模型。

27MHz X-tal 61 产生 27Mhz 的频率。27MHz 频率的误差范围必须是+/-30ppm（27000000+/-810Hz）。到达时间时钟计数器 62 是用于计数 27MHz 频率的脉冲的二进制计数器。Arrival_time_clock(i)是到达时间时钟计数器在时间 t(i)的计数值。

在平滑缓冲器 64 中，当平滑缓冲器不为空时，Rmax 是源数据包到平滑缓冲器的输入位速率。如果平滑缓冲器为空，到平滑缓冲器的输入位速率为 0。

借助于所解释的 MPEG 2 传输流的输出时序，如果当前源数据包的 Arrival_time_stamp 等于 Arrival_time_clock(i)的 LSB（最低有效位）侧的 30 位，则从平滑缓冲器移除源数据包的传输数据包。Rpk 是传输数据包速率的瞬时最大值。不允许平滑缓冲器的溢出。

DVR MPEG 2 传输流的播放器模型的参数与上述 DVR MPEG 2 传输流的记录器模型的参数相同。

图 85 显示了源数据包的句法结构。Transport_packet()是在 ISO/IEC 13818-1 中提供的 MPEG 2 传输流。图 85 中所示的源数据包的句法结构中的 TP_Extra_header 的句法结构示于图 86。作为如图 86 所示的 TP_Extra_header 的句法结构的说明，copy_permission_indicator（复制许可指示符）是代表传输数据包的有效载荷的复制限制的整数。复制限制可以是自由复制、不再复制、复制一次或禁止复制。图 87 显示了 copy_permission_indicator 的值和它指定的模式之间的关系。

copy_permission_indicator 附加于传输数据包的整体。如果输入传输流使

用IEEE1394数字接口进行记录，则copy_permission_indicator的值可以与EMI（加密模式指示符）的值相关。如果输入传输流不使用IEEE1394数字接口进行记录，则copy_permission_indicator的值可以与嵌入在传输数据包中的CCI的值相关。如果模拟信号输入为自编码，则copy_permission_indicator的值可以与模拟信号的CGMS-A的值相关。

Arrival_time_stamp 是具有由下列方程中的 Arrival_time_stamp 指定的值的整数:

Arrival_time_stamp(k)= Arrival_time_clock(k)%230。

经由定义的Clip AV流，Clip AV流必须具有如上面描述所定义的DVR MPEG 2传输流的结构。在Clip AV流中，Arrival_time_clock(i)必须连续增加。如果在Clip AV流中存在系统时间基（STC基准）的断点，则Clip AV流中的Arrival_time_clock(i)必须连续增加。

Clip AV流的开始和结束之间的Arrival_time_clock(i)的最大差值必须是26小时。这个限制保证了，如果在MPEG 2传输流的系统时间基（STC基准）中没有断点，则相同值的PTS（表达时间戳）永远不会出现在Clip AV流中。MPEG 2标准提供了PTS具有233/90000秒（大约26.5小时）的环绕周期。

经由定义Bridge-Clip AV流，Bridge-Clip AV流必须具有如上面描述所定义的DVR MPEG 2传输流的结构。Bridge-Clip AV流必须包括一个到达时间基的断点。在到达时间基的断点的前面和后面的传输流必须遵从编码限制和后面将说明的DVR-STD。

本发明的实施例支持正被编辑的各PlayItem之间的视频-音频无缝连接。各PlayItem之间的无缝连接保证了对播放器/解码器的“连续数据供应”和“无缝解码处理”。“连续数据供应”是以防止缓冲器下溢所需要的位速率保证供应数据给解码器的能力。当保证数据实时特性时，为了使数据能够从盘读出，数据将以足够大的连续数据块为单位进行存储。

“无缝解码处理”意味着播放器在显示记录于盘上的音频视频数据时，解码器的回放输出中不出现停顿或间隔的能力。

现在说明参照无缝连接的PlayItem的AV流。在先的PlayItem和当前的PlayItem的无缝显示是否得到保证，可以从当前PlayItem中定义的connection_condition（连接条件）字段进行验证。对于PlayItem的无缝连接有两种方法，即使用Bridge-Clip的方法和不使用Bridge-Clip的方法。

图 88 显示了在使用 Bridge-Clip 的情况下，在先的 PlayItem 和当前的 PlayItem 之间的关系。图 88 中，播放器读出的流数据用阴影显示。图 88 中的 TS1 由 Clip1(Clip AV 流)的阴影流数据和在 RSPN_arrival_time_discontinuity 之前的阴影流数据组成。

TS1 的 Clip1 的阴影流数据是从解码对应于在先的 PlayItem 的 IN_time（如图 88 中的 IN_time1 所示）的表示单元所需要的流的地址开始到参照 RSPN_exit_from_previous_Clip 的源数据包的流数据。在包含于 TS1 的 Bridge-Clip 的 RSPN_arrival_time_discontinuity 之前的阴影数据流是从 Bridge-Clip 的第一个源数据包开始直到参照 RSPN_arrival_time _discontinuity 的源数据包之前一个的源数据包的流数据。

在图 88 中，TS2 由 Clip2(Clip AV 流)的阴影流数据和接续 Bridge-Clip 的 RSPN_arrival_time_discontinuity 的阴影流数据组成。来自包含于 TS 2 中的 Bridge-Clip 的 RSPN_arrival_time_discontinuity 的阴影流数据是从参照 RSPN_arrival_time_discontinuity 的源数据包开始到 Bridge-Clip 的最后的数据包的流数据。TS2 的 Clip2 的阴影流数据是参照 RSPN_enter_to_curent_Clip 的源数据包开始到解码对应于当前 PlayItem 的 OUT_time（由图 88 中的 OUT_time2 所示）的表示单元所需要的流的地址的流数据。

图 89 显示了在不使用 Bridge-Clip 的情况下，在先的 PlayItem 和当前的 PlayItem 之间的关系。在这种情况下，由播放器读出的流数据用阴影显示。在图 89 中，TS1 由 Clip1（Clip AV 流）的阴影流数据组成。TS1 的 Clip1 的阴影流数据是开始于解码对应于在先的 PlayItem 的 IN_time（如图 89 中的 IN_time1 所示）的表示单元所需要的流的地址、直到 Clip1 的最后的源数据包的数据。

在图 89 中，TS2 是 Clip2（Clip AV 流）的阴影流数据。

TS2 的 Clip2 的阴影流数据是开始于 Clip 2 的第一个源数据包直到解码对应于当前 PlayItem 的 OUT_time（如图 89 中的 OUT_time2 所示）的表示单元所需要的流的地址的流数据。

在图 88 和 89 中，TS1 和 TS2 是源数据包的连续流。接下来，详细说明 TS1 和 TS2 的流供应和它们之间的连接条件。首先，细察无缝连接的编码限制。借助于对传输流的编码结构的限制，包含于 TS1 和 TS2 中的节目数量必须是 1。包含于 TS1 和 TS2 中的视频流数量必须是 1。包含于 TS1 和 TS2 中

的音频流的数量必须是2或更少。包含于TS1和TS2中的音频流的数量必须相等。除上面描述的外，基本流和私有流也可能包含于TS1和/或TS2中。

现在说明对视频位流的限制。图90显示了由画面显示序列指示的典型无缝连接。为了在结合点附近无缝地演示视频流，必须通过再编码结合点附近的Clip的部分流的处理，移除显示于IN_time2(Clip2的IN_time)前面和OUT_time1(Clip1的OUT_time)后面的不需要的画面。

图91显示了使用BridgeSequence(桥接序列)实现无缝连接的实施例。在RSPN_arrival_time_discontinuity之前的Bridge_Clip的视频流包括直到对应于图90的Clip1的OUT_time1的画面的编码视频流。这个视频流连接于在先的Clip1的视频流，并被再编码以形成符合MPEG2标准的基本流。

在RSPN_arrival_time_discontinuity之后的Bridge_Clip的视频流包括接续对应于图90的Clip2的IN_time2的画面的编码视频流。这个视频流的解码可以正确地开始，用于将该视频流连接于下一个Clip2视频流。进行再编码以便形成符合MPEG2标准的唯一连续基本流。为了创建Bridge_Clip，一般需要再编码几个画面，同时其它画面可以从原始Clip中复制。

图92显示了不使用图90所示的实施例中的BridgeSequence而实现无缝连接的实施例。Clip1的视频流包括直到对应于图90的OUT_time1的画面的编码视频流，并被再编码以给出符合MPEG2标准的基本流。用类似的方式，Clip2的视频流由接续与图90的Clip2的IN_time2相关的画面的编码位流组成。这些编码位流已经被再编码以给出符合MPEG2标准的唯一连续基本流。

借助于所说明的视频流的编码限制，TS1和TS2的视频流的帧速率必须相等。TS1的视频流必须终止于sequence_end_code(序列结束代码)。TS2的视频流必须开始于Sequence header(序列标头)、GOP Header（GOP标头），并具有I画面。TS2的视频流必须开始于封闭的GOP。

在位流（帧或半帧（field））中定义的视频表示单元必须在结合点处相连续。在结合点处不允许存在半帧或帧的间隔。在使用3-2下拉编码的情况下，重写“top_field_first”和“repeat_first_field”标志是必要的。另外，可以进行本地再编码以防止产生半帧间隔。

经由所说明的对音频位流的编码限制，TS1的音频采样频率和TS2的音频采样频率必须相等。TS1的音频编码方法和TS2的音频编码方法（例如MPEG1第2层、AC-3、SESF、LPCM和AAC）必须相等。

经由所说明的对 MPEG2 传输流的编码限制，TS1 的音频流的最后的音频帧必须包含具有等于 TS1 的最后显示画面的显示结束时间的显示时刻的各音频采样。TS2 的音频流的第一个音频帧必须包含具有等于 TS2 的第一个显示画面的显示开始时间的显示时刻的音频采样。

在结合点处，不允许在音频表示单元的序列中存在间隔。如图 93 所示，可能有由小于两个音频帧域的音频表示单元的长度定义的交叠。传送 TS2 的基本流的第一个数据包必须是视频数据包。结合点处的传输流必须遵从后面将要说明的 DVR-STD 标准。

经由所说明的对 Clip 和 Bridge-Clip 的限制，不允许在 TS1 或 TS2 中存在到达时间基的中断。

下列的限制只应用于使用 Bridge-Clip 的情况。Bridge-Clip AV 流仅在 TS1 的最后源数据包和 TS2 的第一个源数据包的结合点处具有唯一的到达时间基的断点。ClipInfo()中定义的 SPN_arrival_time_discontinuity 代表断点的地址，该地址必须代表参照 TS2 的第一个源数据包的地址。

参照 BridgeSequenceInfo()中定义的 RSPN_exit_from_previous_Clip 的源数据包可以是 Clip1 中的任何源数据包。该源数据包没有必要位于联合单元的边界。参照 BridgeSequenceInfo()中定义的 RSPN_enter_to_current _Clip 的源数据包可以是 Clip2 中的任何源数据包。该源数据包没有必要位于联合单元的边界。

经由所说明的对 PlayItem 的限制，在先的 PlayItem 的 OUT_time (如图 89 中所示的 OUT_time1)必须代表 TS1 的最后视频表示单元的显示结束时间。当前 PlayItem 的 IN_time (如图 89 中所示的 IN_time2)必须代表 TS2 的第一个表示单元的显示开始时间。

经由参照图 94 所说明的在使用 Bridge-Clip 的情况下对数据分配的限制，必须进行无缝连接以保证由文件系统进行的连续数据供应。这必须通过安排 Bridge-Clip AV 流、连接到 Clip1(Clip 流文件)和 Clip2(Clip 流文件)来实现，以便满足数据分配的规定。

必须选择 RSPN_exit_from_previous_Clip，以便使得在 RSPN_exit_from_previous_Clip 之前的 Clip1（Clip AV 流文件）的部分流将被安排在不小于半个段（fragment）的连续区域内。必须选择 Bridge-Clip AV 流的数据长度，以便使得该数据将被安排在不小于半个段的连续区域内。必须选择

RSPN_enter_to_current_Clip，以使得接续 RSPN_enter_to_current_Clip 的 Clip2（Clip AV 流文件）的部分流将被安排在不小于半个段连续区域内。

经由参照图 95 所说明的在不使用 Bridge-Clip 的无缝连接的情况下的数据分配限制，必须进行无缝连接以保证由文件系统进行的连续数据供应。这必须通过安排 Clip1(Clip AV 流文件)的最后部分和 Clip2(Clip AV 流文件)的第一部分来实现，以便满足对数据分配的规定。

Clip1(Clip AV 流文件)的最后流部分必须被安排在不小于半个段的连续区域内。Clip2(Clip AV 流文件)的第一个流部分必须被安排在不小于半个段的连续区域内。

在每一个具有预定位速率的数字 AV 信号被分成段并记录在盘上的情况下，为了确保所记录的数字 AV 信号能够从记录介质 100 中按预定位速率读出，一个连续记录区域的大小必须满足下述条件:

S*8/（S*8/Rud+Ts）>=Rmax

其中

S: 连续记录区域的最小尺寸[字节]

Ts: 从一个记录区域到下一个记录区域的完全行程的访问时间[秒]

Rud: 从记录介质中读出数据的位速率[位/秒]

Rmax: AV 流的位速率[位/秒]

也就是说，必须排列数据，以便 AV 流中的 S 字节或更多数据可以连续地记录在盘上。

必须排列数据，以便半个段的尺寸为 S 字节或更多。

接下来说明 DVR-STD。此 DVR-STD 是用于对 DVR MPEG2 传输流的产生和校验中的解码处理进行建模的概念性模型。此 DVR-STD 也是用于对如上所述参照相互无缝连接的两个 PlayItem 的 AV 流的产生和校验中的解码处理进行建模的概念性模型。

图 96 显示了 DVR-STD 模型。作为组成的元件，图 96 所示的模型包括 DVR MPEG 2 传输流播放器模型。符号 n、Tbn、Mbn、Ebn、Tbsys、Bsys、Rxn、Rbxn、Rxsys、Dn、Dsys、On 和 P9(k)与 ISO/IEC 13818-1 的 T-STD 中定义的相同，其中 n 是基本流的索引号，TBn 是基本流 n 的传输缓冲器。

MBn 是基本流 n 的多路复用缓冲器并仅为视频流而存在。EBn 是基本流 n 的基本流缓冲器并仅为视频流而存在。TBsys 是用于正在被解码的节目的系

统信息的系统目标解码器中的主缓冲器。Rxn 是从 TBn 移除数据的传送速率。Rbxn 是从 MBn 移除 PES 数据包有效载荷的传送速率并仅为视频流而存在。

Rxsys 是从 TBsys 移除数据的传送速率。Dn 是基本流 n 的解码器。Dsys 是与正在被解码的节目的系统信息有关的解码器。On 是视频流 n 的再排序缓冲器。Pn(k)是基本流的第 k 表示单元。

现在说明 DVR-STD 的解码过程。在唯一 DVR MPEG 2 传输流正在被再现期间，由源数据包的 arrival_time_stamp 确定将传输数据包输入给 TB1、TBn 或 Tbsys 的时刻。对 TB1、MB1、EB1、TBn、Bn、Tbsys 和 Bsys 的缓冲操作的规定与 ISO/IEC 13818-1 提供的 T-STD 中的定义相同，而对决定和显示操作的规定也与 ISO/IEC 13818-1 提供的 T-STD 相同。

现在说明在无缝连接的 PlayList 正在被再现期间的解码处理。这里，说明参照无缝连接的 PlayItem 的两个 AV 流的再现。在下面的说明中，说明例如图 88 中所示的 TS1 和 TS2 的再现。TS1 和 TS2 分别是在先流和当前流。

图 97 显示了用于当从给定的 AV 流（TS1）向与其无缝连接的下一个 AV 流（TS2）转换时进行输入、解码和显示传输数据包的时序图。在从预设的 AV 流（TS1）向与其无缝连接的下一个 AV 流（TS2）的转换期间，TS2 的到达时间基的时间轴与 TS1 的到达时间基的时间轴（由图 97 中的 ATC1 指示）不同。

再有，TS2 的系统时间基的时间轴（由图 97 中的 STC2 指示）与 TS1 的系统时间基的时间轴（由图 97 中的 STC1 指示）不同。视频显示需要无缝地连续，可是在表示单元的显示尖峰上可能存在交叠。

现在说明对 DVR-STD 的输入时刻。在直到时间 T1 的期间，即直到将最后的视频数据包输入到 DVR-STD 的 TB1，对 DVR-STD 的 TB1、TBn 或 TBsys 的缓冲器的输入时刻由 TS1 的到达时间基的 arrival_time_stamp 确定。

TS1 的剩余数据包必须以 TS_recording_rate(TS1)的位速率被输入 TBn 的缓冲器或 DVR-STD 的 TBsys 中。TS_recording_rate(TS1)是对应于 Clip1 的 ClipInfo()中定义的 TS_recording_rate 的值。TS1 的最后字节被输入缓冲器的时间是时间 T2。因此，在时间 T1 和时间 T2 之间的时间期间，源数据包的 arrival_time_stamp 减小（discount）。

如果 N1 是接着 TS1 的最后视频数据包的 TS1 的传输数据包的字节数，则从时间 T1 到时间 T2 的时间 DT1 是 N1 字节以 TS_recording_rate(TS1)的位

速率完全输入所必需的时间，并根据下列方程计算：

DT1=T2-T1=N1/TS_recording_rate。

从时间 T1 到时间 T2 的时间期间，RXn 和 Rxsys 两者的值都被变为 TS_recording_rate(TS1)的值。除了这条，缓冲操作与 T-STD 的相同。

在时间 T2，到达时间时钟计数器被复位为 TS2 的第一个源数据包的 arrival_time_stamp 的值。对 DVR-STD 的 TB1、TBn 或 TBsys 的缓冲器的输入时刻由 TB2 的源数据包的 arrival_time_stamp 确定。RXn 和 RXsys 两者的值都被变为 T-STD 中定义的值。

经由所说明的附加音频缓冲和系统数据缓冲，音频解码器和系统解码器除了 T-STD 中定义的缓冲量外，需要具有附加缓冲量（相当于 1 秒钟的数据量），以便允许从时间 T1 到时间 T2 的域的输入数据。

经由所说明的视频表示时刻，视频表示单元上的显示必须连续地经过结合点，即没有间隔。应注意到，STC1 是 TS1 的系统时间基的时间轴（如图 97 中的 STC1 所指示），而 STC2 是 TS2 的系统时间基的时间轴（如图 97 中的 STC2 所指示）。正确地，STC2 开始于 TS2 的第一个 PCR 已经被输入到 T-STD 的时间。

STC1 和 STC2 之间的偏移如下确定：$PTS1_{end}$ 是对应于 TS2 的最后视频表示单元的 STC1 上的 PTS。$PTS2_{start}$ 是对应于 TS2 的第一个视频表示单元的 STC2 上的 PTS，Tpp 是 TS1 的最后视频表示单元的显示时间周期。两个系统时间基之间的偏移 STC_delta 根据下列方程计算：$STC\_delta = PTS1_{end} + Tpp - PTS2_{start}$。

经由所说明的音频表示时刻，在音频表示单元的显示时间时刻可能有交叠，该交叠小于 0 到 2 个音频帧（见图 97 中所示的“音频交叠”）。将选择音频采样的指示，并且音频表示单元的显示与结合点后面的校正时间基的再同步在播放器上设置。

经由所说明的 DVR-STD 的系统时间时钟，在时间 T5 显示 TS1 的最后音频表示单元。系统时间时钟可以在时间 T2 和时间 T5 之间交叠。在这个时间域期间，DVR-STD 在旧时间基（STC1）的值和新时间基（STC2）的值之间切换系统时间时钟。STC2 的值可以根据下列公式计算：STC2=STC1-STC_delta。

现在说明缓冲连续性。当第一个视频数据包的第一个字节到达 DVR-STD

的 TB1 时，STC11video_end 是系统时间基 STC2 上的 STC 的值。当第一个视频数据包的第一个字节到达 DVR-STD 的 TB1 时，STC22video_start 是系统时间基 STC2 上的 STC 的值。STC21video_end 是按照系统时间基 STC2 的 STC2 上的值计算的 STC11video_end 的值。STC21video_end 按照下列方程计算：

STC21video_end= STC11video_end- STC_delta。

为了遵从 DVR-STD，下列两个条件必须满足：第一，在 TB1 处的 TS2 的第一个视频数据包的到达时刻必须满足下列不等式：

STC22video_start> STC21video_end+ΔT1。

如果有必要以满足上述不等式的方式再编码和/或多路复用 Clip1 和/或 Clip2 的部分流，则这个再编码或多路复用可以适当地得到执行。

第二，由从系统时间基的时间轴上的 TS2 的视频数据包的输入跟随的、从相同时间轴上的 STC1 和 STC2 映射的、从 TS1 的视频数据包的输入，不必溢出或下溢视频缓冲器。

如果上面的句法、数据结构和规则被用作基础，则可以适当地管理记录于记录介质上的数据内容或再现信息，使用户能够在再现时确认记录于记录介质上的数据内容，或相当容易地再现期望的数据。

在上述的实施例中，将 MPEG 2 传输流作为多路复用的流示例。但是，这仅仅用于示范，而且 MPEG 2 节目流 DSS 或美国的 DirecTV Service（商标）中使用的传输流也可以用作所述多路复用的流。

图 98 是用于说明准备 RealPlayList 的流程图。参照图 1 的方框图所示的记录和/或再现设备 1。在步骤 S10，控制器 23 记录 Clip AV 流。在步骤 S11，控制器 23 形成由涵盖上述 Clip 的可能的总播放范围的 PlayItem 构成的 PlayList()。如果在 Clip 中存在 STC 断点，并且 PlayList()由两个或更多 PlayItem 构成，则还确定各个 PlayItem 之间的 connection_condition。

在步骤 S12，控制器 23 形成 UIAppInfoPlayList()。在步骤 S13，控制器 23 形成 PlayListMark。在步骤 S14，控制器 23 形成 MakersPrivateData。在步骤 S15，控制器 23 记录 Real PlayList 文件。以这种方式，每一次新记录一个 Clip AV 流文件，新创建一个 Real PlayList 文件。

图 99 是用于说明定制具有 Bridge Sequence(桥接序列)的 VirtualPlayList 的流程图。在步骤 S20，通过用户接口指定记录在盘上的一个 Real PlayList 的再现。从 Real PlayList 的再现范围，由 IN 和 OUT 指定的再现域通过用户

接口指定。

在步骤 S21，控制器 23 验证由用户指定再现范围的操作是否已经全部完成。如果证实已经完成指定操作，则控制器 23 前进到步骤 S22，否则，控制器 23 返回到步骤 S20，重复随后的处理。

在步骤 S22，用户通过用户接口确定两个连续地再现的 PlayItem 之间的连接条件（connection_condition），或者控制器 23 确定该连接条件。在步骤 S23，控制器 23 形成用于无缝地连接的 PlayItem 的桥接序列。在步骤 S24，控制器 23 形成并记录 Virtual PlayList 文件。

图 100 是用于说明在步骤 S23 中的详细处理的流程图。在步骤 S31，控制器 23 重新编码和重新多路复用显示在临时前向侧的 PlayItem 的 OUT 点一方上的 AV 流。在步骤 S32，控制器 23 重新编码和重新多路复用代表紧接该 PlayItem 的 PlayItem 的 IN 点一方上的 AV 流。

在步骤 S33，控制器 23 确定 RSPN_exit_from_previous_Clip 的值，以便满足用于连续数据供应的数据分配条件。也就是说，必须选择 RSPN_exit_from_previous_Clip，以便将 RSPN_exit_from_previous_Clip 前面的 Clip AV 流文件的流部分记录在不小于前面所述的记录介质的半个段的连续区域中（见图 91 和 94）。

在步骤 S34，控制器 23 确定 RSPN_enter_to_current_Clip 的值，以便满足用于连续数据供应的数据分配条件。也就是说，必须选择 RSPN_enter_to_current_Clip，以便将紧接 RSPN_enter_to_current_Clip 的 Clip AV 流文件的流部分记录在不小于前面所述的记录介质的一个段的连续区域中（见图 91 和 94）。

在步骤 S35，控制器 23 形成 Bridge_Clip AV 流文件，以便满足用于连续数据供应的数据分配条件。也就是说，如果在步骤 S31 和 S32 的处理中准备的数据量少于前面所述的半个段，则从初始的 Clip 中复制数据，以便准备 Bridge_Clip（见图 91 和 94）。

尽管步骤 S33 至 S35 的处理操作是按照顺序说明的，但是这些处理操作也可以不按照上述顺序地执行，或者可以同时执行，这是因为这些处理操作是可以相反的。

在步骤 S36，控制器 23 形成桥接序列数据库。在步骤 S37，控制器 23 记录 Bridge_Clip AV 流文件及其 Clip Information 文件。以这种方式，由用户

从记录在盘上的 Real PlayList 的再现范围中选择一个或多个 PlayItem。形成用于使两个 PlayItem 无缝连接的桥接序列，并将分组在一起的一个或多个 PlayItem 的集合记录为一个 Virtual PlayList。

图 101 是用于说明再现 PlayList 的流程图。在步骤 S41，控制器 23 通过用户接口获取 Info.dvr、Clip Information 文件、PlayList 文件和 thumbnail 信息以准备 GUI 画面、以及一列记录在盘上的 PlayList 以在 GUI 上显示如此形成的画面。

在步骤 S42，控制器 23 根据每一个 PlayList 的 UIAppInfoPlayList()，呈现用于说明 GUI 画面上的 PlayList 的信息。在步骤 S43，用户通过用户接口从 GUI 画面中指令一个 PlayList 的再现。在步骤 S44，控制器 23 从 IN_time 的 PTS 和当前的 PlayItem 的 STC_sequence_id 中获取具有临时在先并最接近 IN_time 的入口点（entry point）的源数据包。

在步骤 S45，控制器 23 从具有上述入口点的源数据包序号中读出 AV 流的数据，以便将如此读出的数据发送给解码器。在步骤 S46 中，如果存在临时位于当前的 PlayItem 之前的 PlayItem，则控制器 23 根据 connection_condition 执行相互连接在先的 PlayItem 和当前的 PlayItem 的处理。如果这些 PlayItem 被无缝地相互连接，则根据 DVR-STD 解码方法将 AV 流解码。

在步骤 S47，控制器 23 命令 AV 解码器 27 从 IN_time 的 PTS 的画面开始显示。在步骤 S48，控制器 23 命令 AV 解码器 27 继续解码 AV 流。在步骤 S49，控制器 23 验证当前所显示的画面是否是 OUT_time 的 PTS 的画面。如果证实当前所显示的画面不是 OUT_time 的 PTS 的画面，则控制器 23 前进到步骤 S50 显示画面，之后控制器返回步骤 S48，重复随后的处理。

如果在步骤 S49 证实当前所显示的画面是 OUT_time 的 PTS 的画面，则控制器 23 前进到步骤 S51，在此控制器 23 验证当前的 PlayItem 是否是 PlayList 中的最后一个。如果证实当前的 PlayItem 不是最后一个，则控制器 23 返回到步骤 S44，重复随后的处理。如果证实当前的 PlayItem 是最后一个，则结束该 PlayList 的再现。

以这种方式，再现由用户指定的欲再现的一个 PlayList 文件。

根据上述句法、数据结构和规则，如上所述，可以恰当地管理记录在记录介质上的数据内容、播放信息等等，以便使用户恰当地确认记录在记录介质上的数据内容，从而极其方便地再现所期望的数据。

上述操作序列不仅可以通过硬件执行，而且也可以由软件来实现。如果所述操作序列由软件来执行，则该软件从记录介质中安装到用于装载构成该软件的程序的专用硬件的计算机中，或者例如安装到能够根据所安装的各种程序执行各种功能的普通目的的个人计算机中。

图 102 示例示出普通目的的计算机的内部结构。该个人计算机的 CPU（中央处理单元）201 根据存储在 ROM（只读存储器）202 中的程序执行各种处理操作。在 RAM（随机存取存储器）203 中适当地存储有 CPU 201 执行各种处理操作所需要的数据或程序。输入/输出接口 205 与键盘、鼠标等构成的输入单元 206 相连接，以便将输入到输入单元 206 的信号输出到 CPU 201。输入/输出接口 205 还与显示器、扬声器等构成的输出单元 207 相连接。

输入/输出接口 205 还与例如由硬盘构成的记录单元 208、以及适宜通过诸如因特网的网络与外部设备进行数据交换的通信单元 209 相连接。驱动器 210 用于向或从诸如磁盘 221、光盘 222、磁光盘 223、或半导体存储器 224 的记录介质中写或读数据。

记录介质不仅包括所分发的用于提供程序给计算机和用户的数据包型介质，比如包括软盘在内的其上携载程序的磁盘 221、包括 CD-ROM（只读光盘）或 DVD（数字通用盘）的光盘 222、包括小型盘的磁光盘 223、或半导体存储器 224，而且包括内置于计算机中提供给用户的、包括用于携载程序的 ROM 202 和存储器 208 在内的硬盘，如图 108 所示。

在本说明书中，通过介质提供的程序的步骤不仅包括按照所说明的序列执行的按时间顺序排列的处理，而且包括不按时间顺序、而是并行或独立执行的处理。

此外，在本说明书中，系统指的是包括多个部件单元的整个设备。

工业可应用性

在根据本发明的信息处理设备、方法及程序中，如果命令连续地执行从第一 AV 流到第二 AV 流的再现，则生成由第一 AV 流的预设部分和第二 AV 流的预设部分构成的第三 AV 流，并且当再现从第一 AV 流切换到第二 AV 流时再现第三 AV 流；同时生成地址信息，作为与第三 AV 流相关的信息，所述地址信息包括在将再现从第一 AV 流切换到第三 AV 流的时刻关于第一 AV 流的源数据包的地址的信息、以及在将再现从第三 AV 流切换到第二 AV 流的时

刻关于第二 AV 流的源数据包的地址的信息。因此，可以实现保持分开记录的 AV 流连续性的再现。

在根据本发明的信息处理设备、方法及程序中，从记录介质上读出第一 AV 流、第二 AV 流、或第三 AV 流；并从记录介质上读出地址信息作为与第三 AV 流相关的信息，所述地址信息包括在将再现从第一 AV 流切换到第三 AV 流的时刻关于第一 AV 流的源数据包的地址的信息、以及在将再现从第三 AV 流切换到第二 AV 流的时刻关于第二 AV 流的源数据包的地址的信息；以及根据读出的与第三 AV 流相关的信息，在再现进行时，将再现从第一 AV 流切换到第三 AV 流、以及从第三 AV 流切换到第二 AV 流。因此，可以实现保持分开记录的 AV 流连续性的再现。

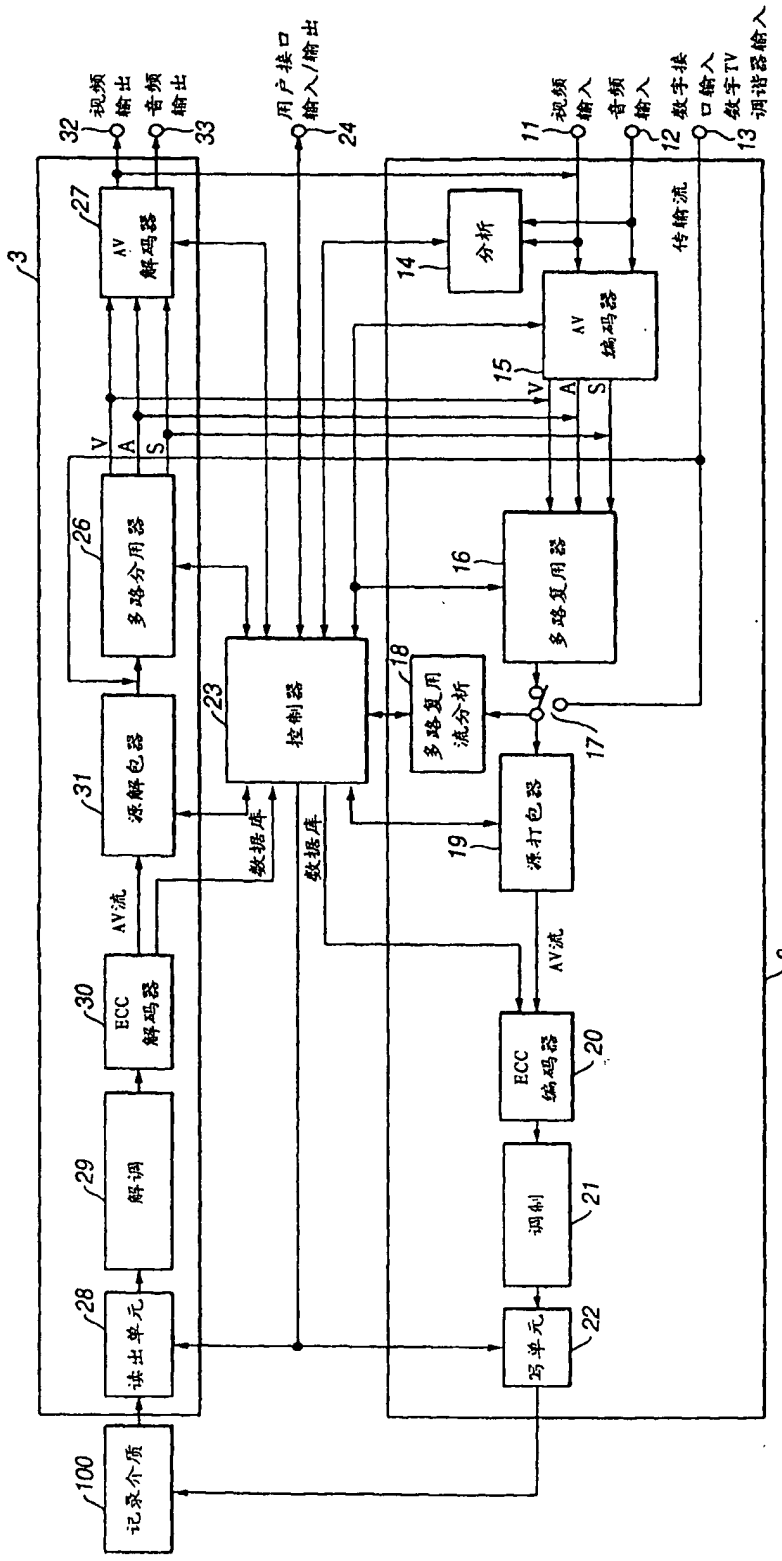

3
2
32 视频输出
33 音频输出
24 用户接口输入/输出
11 视频输入
12 音频输入
13 数字接口输入 数字TV调谐器输入
27 AV解码器
14 分析
AV编码器
15
V
A
S
传输流
26 多路分用器
16 多路复用器
23 控制器
18 多路复用流分析
17
31 源解包器
19 源打包器
数据库
数据库
AV流
AV流
30 ECC解码器
ECC编码器 20
29 解调
调制 21
28 读出单元
写单元 22
100 记录介质

图 1

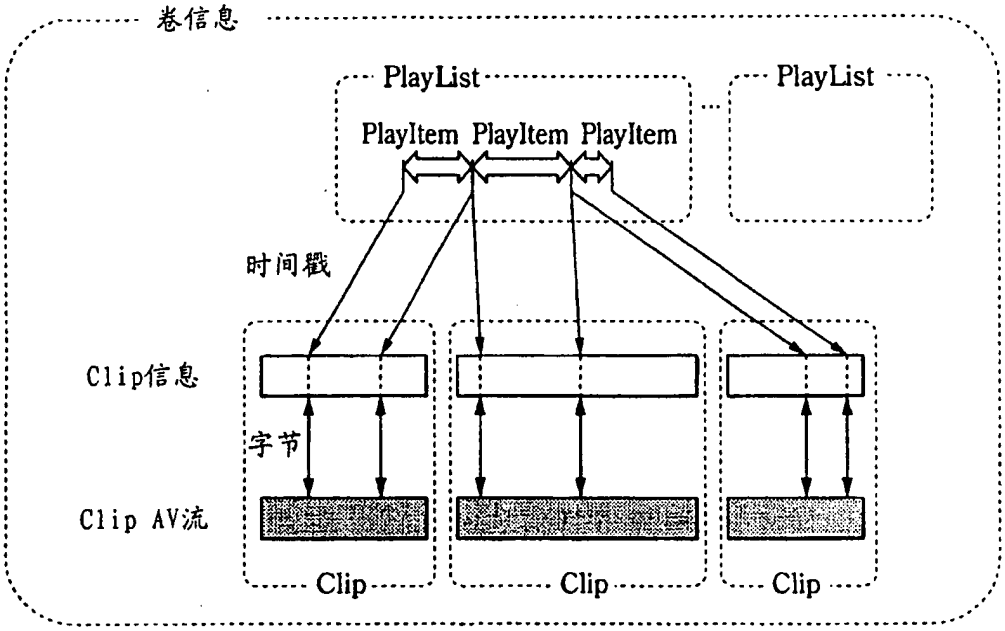

卷信息
PlayList
PlayList
PlayItem PlayItem PlayItem
时间戳
Clip信息
字节
Clip AV流
Clip
Clip
Clip

图 2

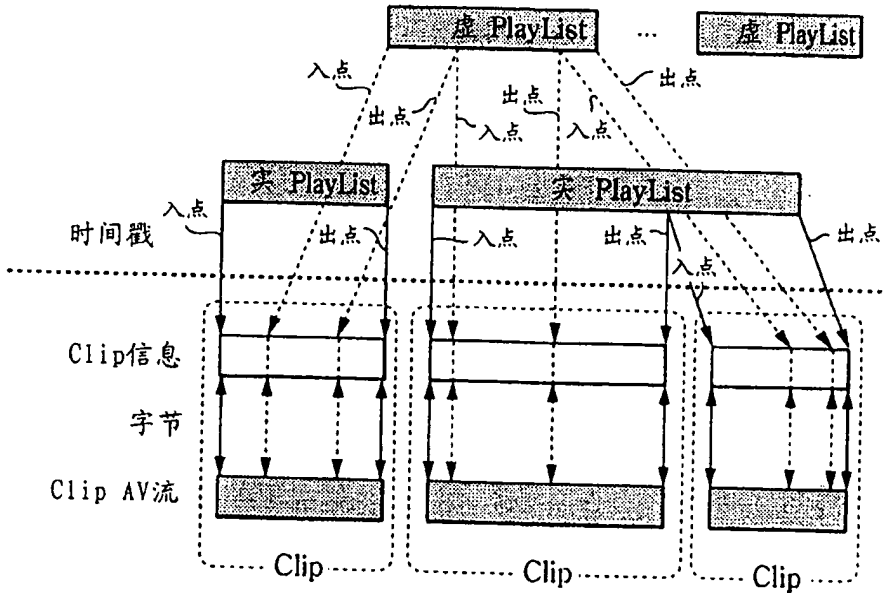

虚 PlayList
...
虚 PlayList
入点
出点
出点
入点
出点
入点
实 PlayList
实 PlayList
入点
时间戳
出点
入点
出点
入点
出点
Clip信息
字节
Clip AV流
Clip
Clip
Clip

图 3

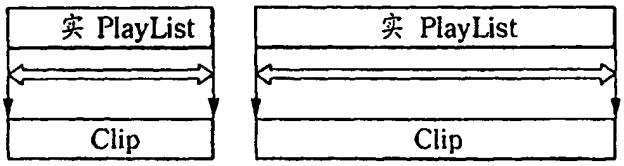

实 PlayList
Clip
实 PlayList
Clip

图 4A

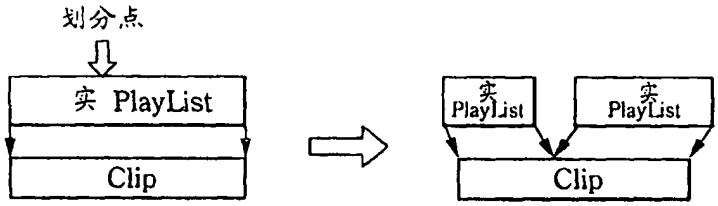

划分点
实 PlayList
Clip
实 PlayList
实 PlayList
Clip

图 4B

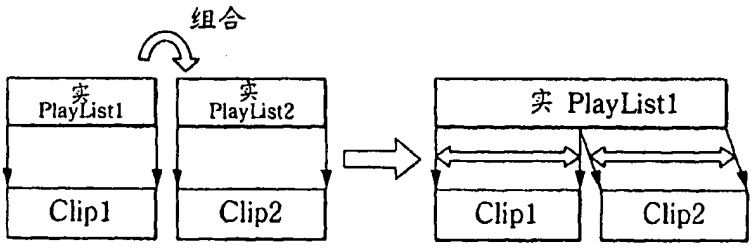

组合
实 PlayList1
实 PlayList2
Clip1
Clip2
实 PlayList1
Clip1
Clip2

图 4C

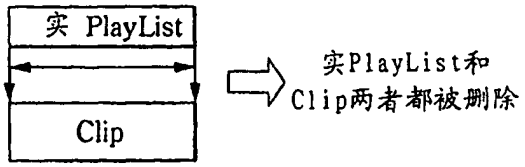

实 PlayList
Clip
实PlayList和
Clip两者都被删除

图 5A

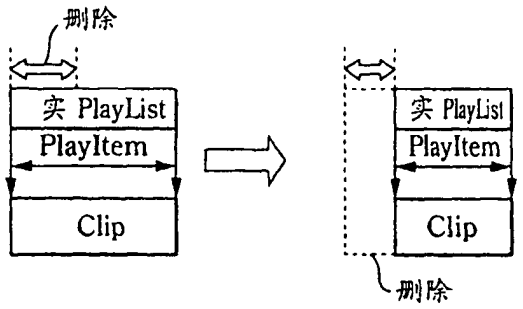

删除
实 PlayList
PlayItem
Clip
实 PlayList
PlayItem
Clip
删除

图 5B

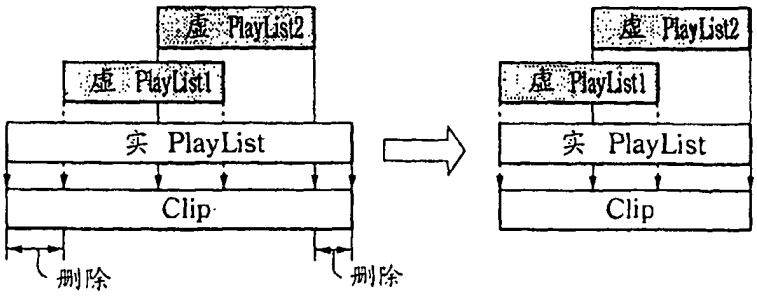

虚 PlayList2
虚 PlayList1
实 PlayList
Clip
删除
删除
虚 PlayList2
虚 PlayList1
实 PlayList
Clip

图 5C

图 6A

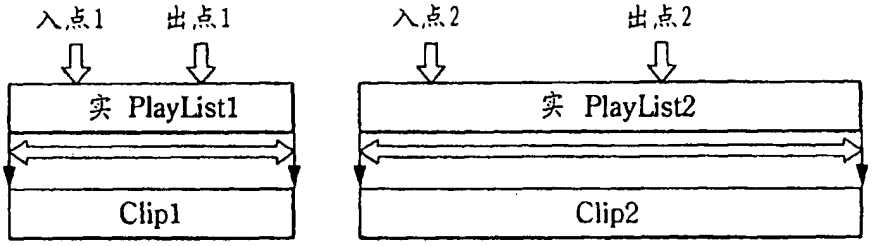
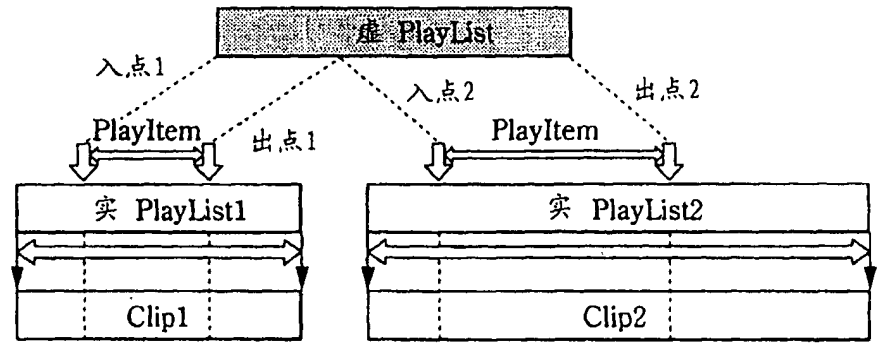

入点1
出点1
入点2
出点2
实 PlayList1
实 PlayList2
Clip1
Clip2

图 6B


虚 PlayList
入点1
出点1
入点2
出点2
PlayItem
PlayItem
实 PlayList1
实 PlayList2
Clip1
Clip2

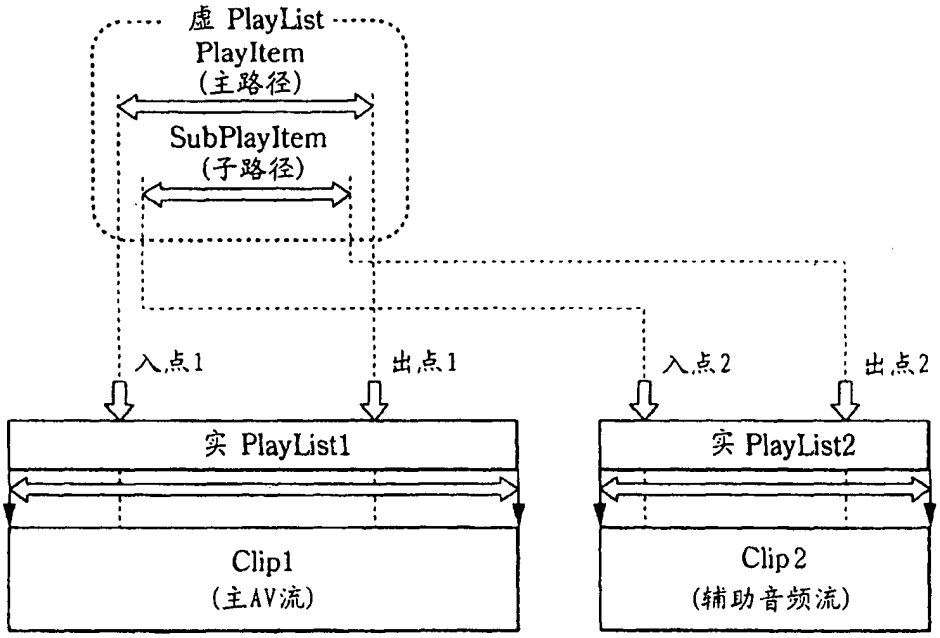

虚 PlayList
PlayItem
(主路径)
SubPlayItem
(子路径)
入点1
出点1
入点2
出点2
实 PlayList1
实 PlayList2
Clip1
(主AV流)
Clip2
(辅助音频流)

图 7

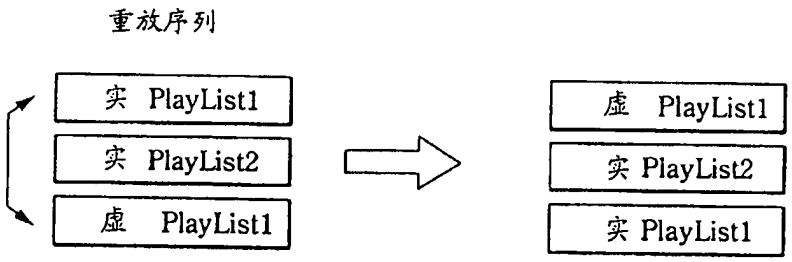

重放序列
实 PlayList1
实 PlayList2
虚 PlayList1
虚 PlayList1
实 PlayList2
实 PlayList1

图 8

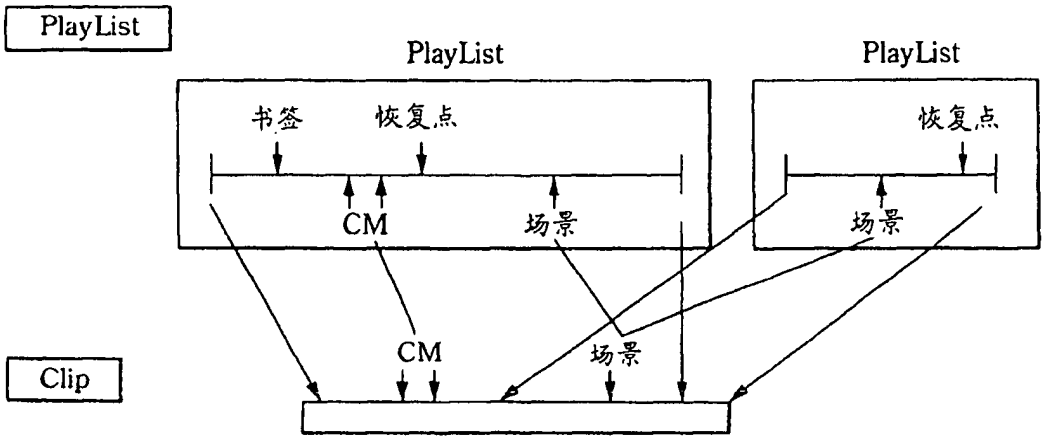


图 9

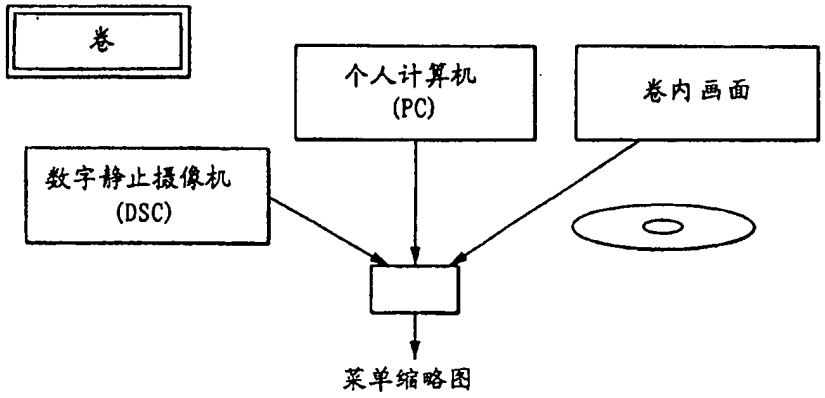

卷
个人计算机
(PC)
卷内画面
数字静止摄像机
(DSC)
菜单缩略图

图 10

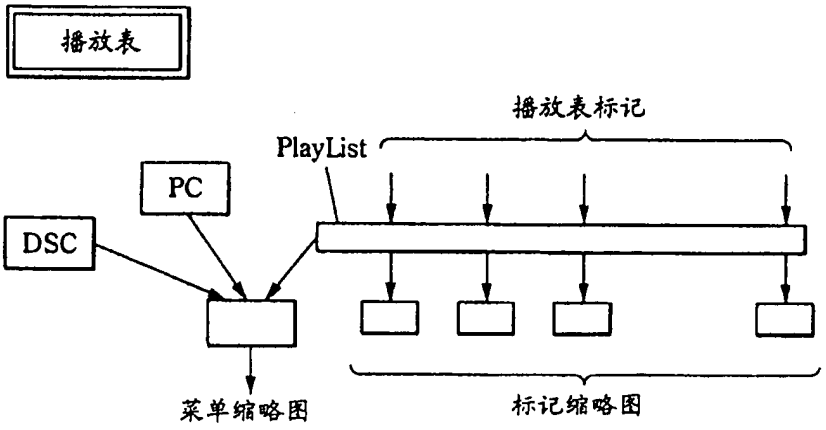

播放表
播放表标记
PlayList
PC
DSC
菜单缩略图
标记缩略图

图 11

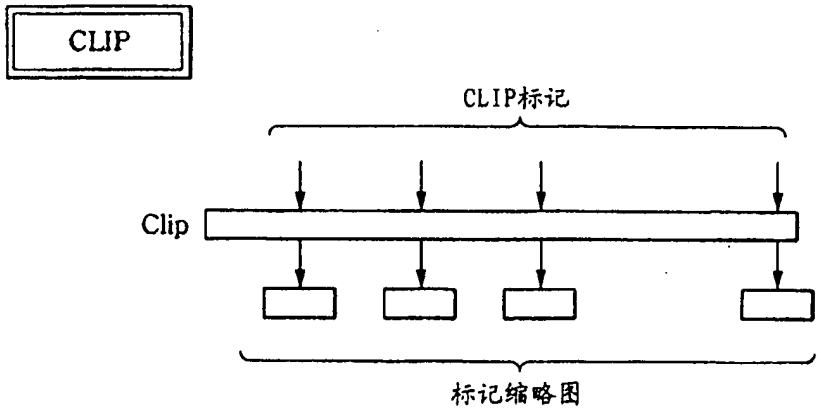

CLIP
CLIP标记
Clip
标记缩略图

图 12

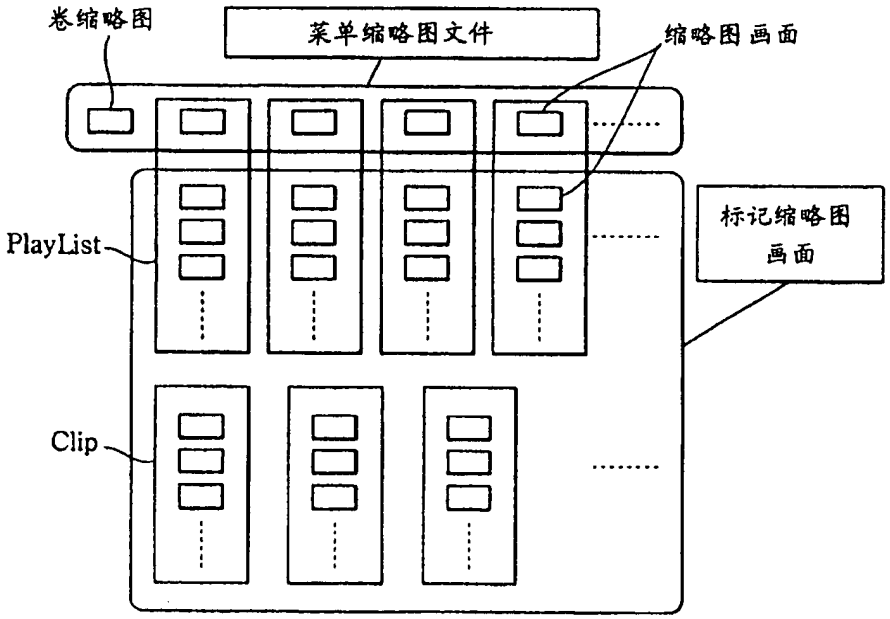

卷缩略图
菜单缩略图文件
缩略图画面
PlayList
标记缩略图
画面
Clip

图 13

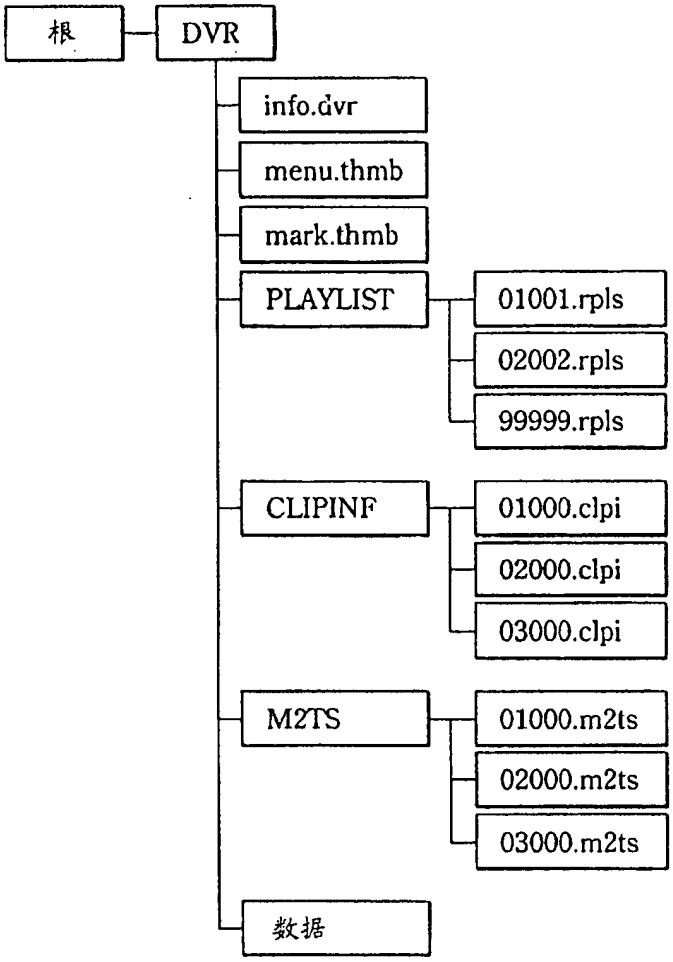

根
DVR
info.dvr
menu.thmb
mark.thmb
PLAYLIST
01001.rpls
02002.rpls
99999.rpls
CLIPINF
01000.clpi
02000.clpi
03000.clpi
M2TS
01000.m2ts
02000.m2ts
03000.m2ts
数据

图 14

句法	字节数	缩写
info.dvr {		
TableOfPlayLists_Start_address	32	uimsbf
MakersPrivateData_Start_address	32	uimsbf
保留	192	bslbf
DVRVolume()		
for (i=0;i<N1;i++){		
padding_word	16	bslbf
}		
TableOfPlayLists()		
for (i=0;i<N2;i++){		
padding_word	16	bslbf
}		
MakersPrivateData()		
}		

图 15

句法	字节数	缩写
DVRVolume(){		
version_number	8*4	bslbf
length	32	uimsbf
ResumeVolume()		
UIAppInfoVolume()		
}		

图 16

句法	字节数	缩写
ResumeVolume(){		
保留	15	bslbf
valid_flag	1	bslbf
resume_PlayList_name	8*10	bslbf
}		

图 17

句法	字节数	缩写
UIAppInfoVolume(){		
character_set	8	bslbf
name_length	8	uimsbf
Volume_name	8*256	bslbf
保留	15	bslbf
Volume_protect_flag	1	bslbf
PIN	8*4	bslbf
ref_thumbnail_index	16	uimsbf
reserved_for_future_use	256	bslbf
}		

图 18

值	字符字母编码
0x00	保留
0x01	ISO/IEC 646 (ASCII)
0x02	ISO/IEC 10646-1 (单一码)
0x03-0xff	保留

图 19

句法	字节数	缩写
TableOfPlayLists(){		
version_number	8*4	bslbf
length	32	uimsbf
number_of_PlayLists	16	uimsbf
for (i=0; i<*number_of_PlayLists*; i++){		
PlayList_file_name	8*10	bslbf
}		
}		

图 20

句法	字节数	缩写
TableOfPlayLists(){		
version_number	8*4	bslbf
length	32	uimsbf
number_of_PlayLists	16	uimsbf
for (i=0; i<*number_of_PlayLists*; i++){		
PlayList_file_name	8*10	bslbf
UIAppInfoPlayList()		
}		
}		

图 21

句法	字节数	缩写
MakersPrivateData(){		
version_number	8*4	bslbf
length	32	uimsbf
if (length !=0){		
mpd_blocks_start_address	32	uimsbf
number_of_maker_entries	16	uimsbf
mpd_block_size	16	uimsbf
number_of_mpd_blocks	16	uimsbf
保留	16	bslbf
for (i=0; i<*number_of_maker_entries*; i++){		
maker_ID	16	uimsbf
maker_model_code	16	uimsbf
start_mpd_block_number	16	uimsbf
保留	16	bslbf
mpd_length	32	uimsbf
}		
stuffing_bytes	8*2*L1	bslbf
for(j=0; j<*number_of_mpd_blocks*; j++){		
mpd_block	mpd_block_size*1024*8	
}		
}		
}		

图 22

句法	字节数	缩写
xxxxx.rpls / yyyyy.vpls {		
PlayListMark_Start_address	32	uimsbf
MakersPrivateData_Start_address	32	uimsbf
保留	192	bslbf
PlayList()		
for (i=0;i<N1;i++){		
padding_word	16	bslbf
}		
PlayListMark()		
for (i=0;i<N2;i++){		
padding_word	16	bslbf
}		
MakersPrivateData()		
}		

图 23

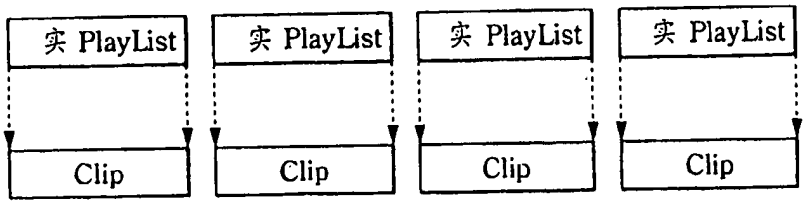


图 24A

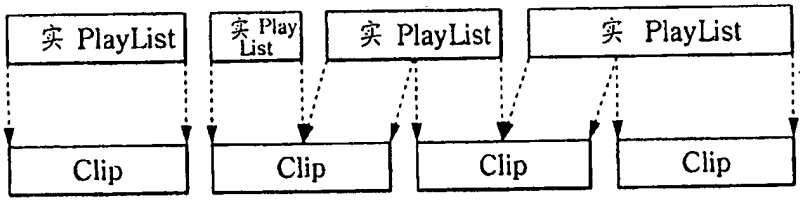


图 24B

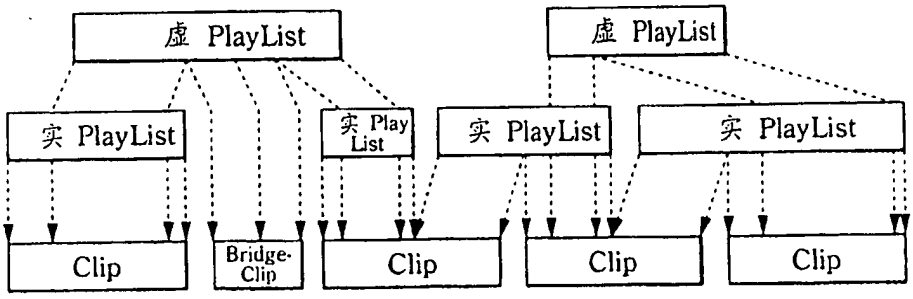


图 24C

句法	字节数	缩写
PlayList(){		
version_number	8*4	bslbf
length	32	uimsbf
PlayList_type	8	uimsbf
CPI_type	1	bslbf
保留	7	bslbf
UIAppInfoPlayList()		
number_of_PlayItems //主路径	16	uimsbf
if (<*Vertual PlayList*>){		
number_of_SubPlayItems //子路径	16	uimsbf
}else{		
保留	16	bslbf
}		
for (*PlayItem_id=0;* *PlayItem_id<nymber_of_PlayItems;* *PlayItem_id++*){		
PlayItem() //主路径		
}		
if (<*Virtual PlayList*>){		
if (CPI_type==0 && PlayList_type==0){		
for (i=0; i<*number_of _SubPlayItems*; i++)		
SubPlayItem() //子路径		
}		
}		
}		

图 25

PlayList_type	含义
0	AV记录的播放表; 在该播放表中引用的所有CLIPS一定包含一个 或多个视频流
1	音频记录的播放表; 该播放表中引用的所有CLIPS一定包含一个 或多个音频流并且一定不包含视频流
2-255	保留

图 26

句法	字节数	缩写
UIAppInfoPlayList2()		
character_set	8	bslbf
name_length	8	uimsbf
PlayList_name	8*256	bslbf
保留	8	bslbf
record_time_and_date	4*14	bslbf
保留	8	bslbf
duration	4*6	bslbf
valid_period	4*8	bslbf
maker_id	16	uimsbf
maker_code	16	uimsbf
保留	11	bslbf
playback_control_flag	1	bslbf
write_protect_flag	1	bslbf
is_played_flag	1	bslbf
archive	2	bslbf
ref_thumbnail_index	16	uimsbf
reserved_for_future_use	256	bslbf
}		

图 27

write_protect_flag	含义
0b	PlayList能被自由地删除
1b	除write_protect_flag之外PlayList内容 应不能被擦除和改变

图 28A

is_played_flag	含义
0b	自从其记录开始，PlayList没有被再现
1b	自从其记录开始，PlayList被再现一次

图 28B

文档	含义
00b	没有定义意义
01b	原始的
10b	拷贝
11b	保留

图 28C

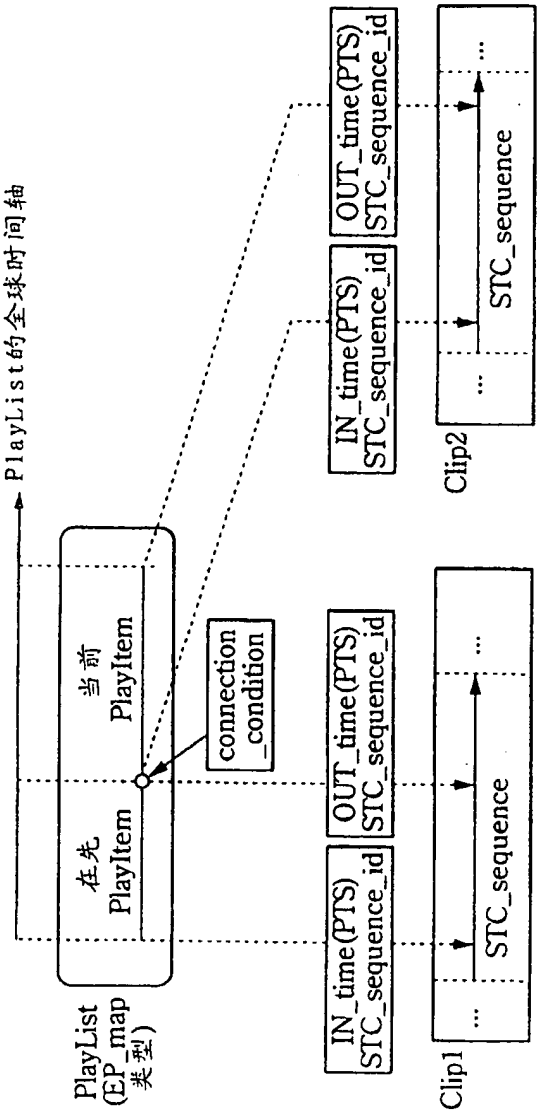

PlayList的全球时间轴
PlayList
(EP_map
类型)
在先
PlayItem
当前
PlayItem
connection
_condition
IN_time(PTS)
STC_sequence_id
OUT_time(PTS)
STC_sequence_id
Clip1
STC_sequence
IN_time(PTS)
STC_sequence_id
OUT_time(PTS)
STC_sequence_id
Clip2
STC_sequence

图 29

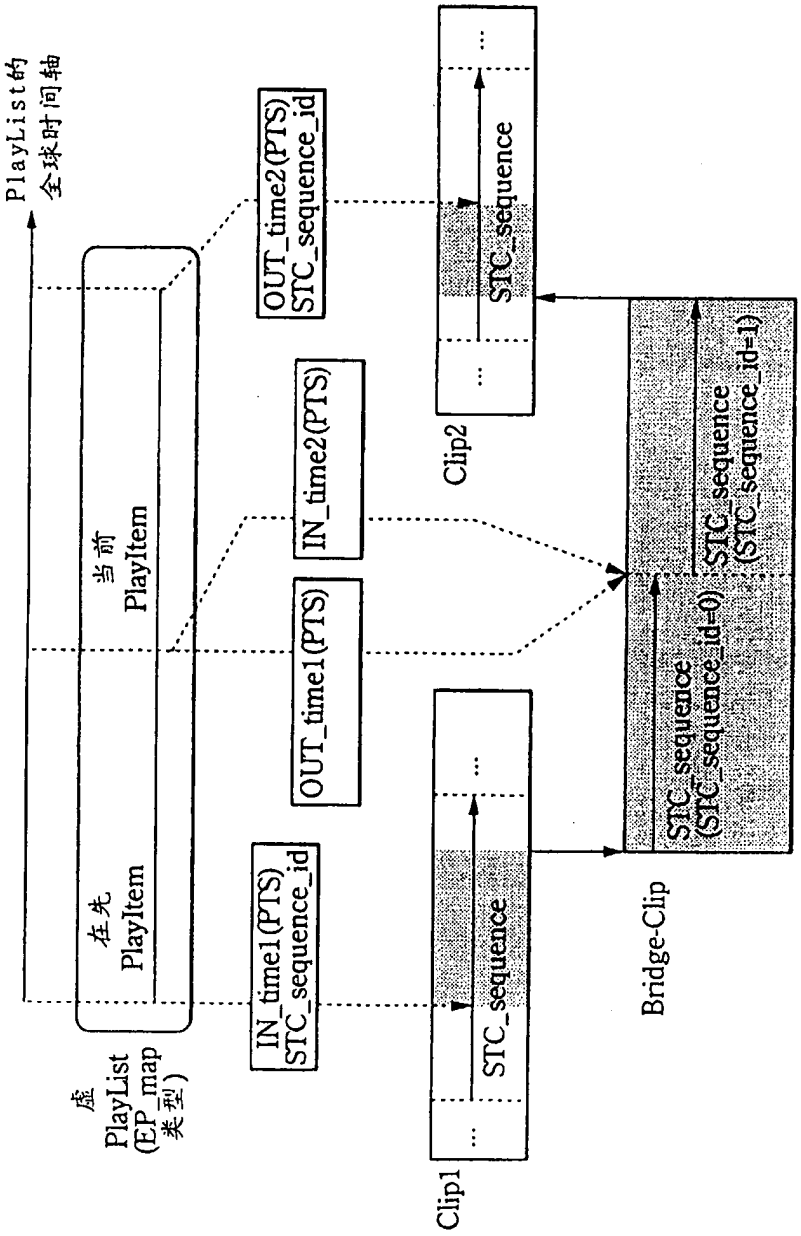

PlayList的
全球时间轴
虚
PlayList
(EP_map
类型)
在先
PlayItem
当前
PlayItem
IN_time1(PTS)
STC_sequence_id
OUT_time1(PTS)
IN_time2(PTS)
OUT_time2(PTS)
STC_sequence_id
Clip1
STC_sequence
Clip2
STC_sequence
Bridge-Clip
STC_sequence
(STC_sequence_id=0)
STC_sequence
(STC_sequence_id=1)

图 30

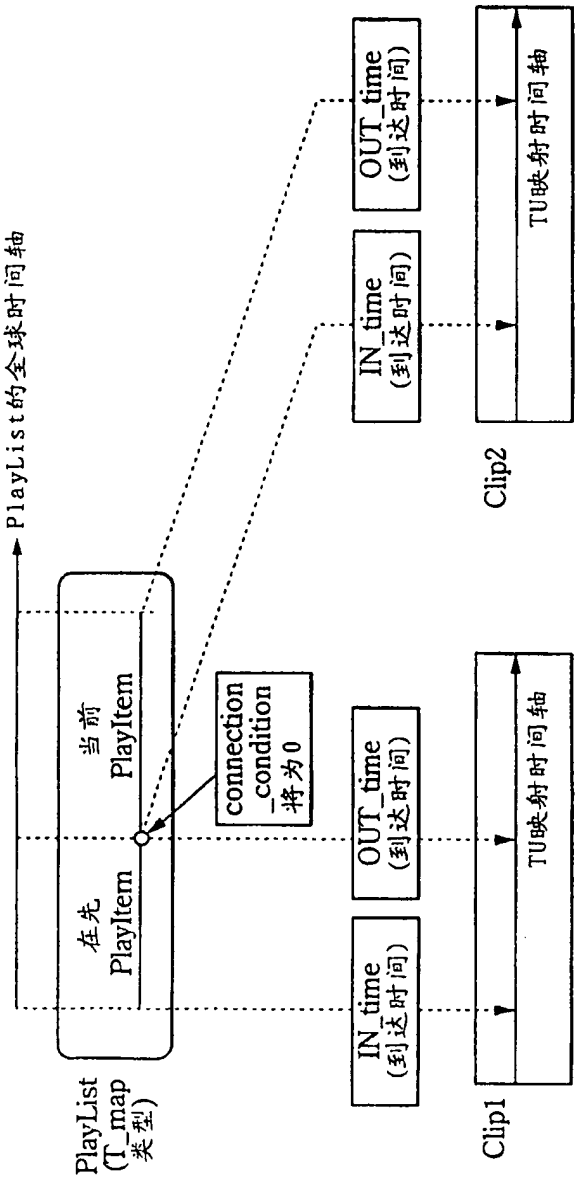

PlayList的全球时间轴
PlayList
(T_map
类型)
在先
PlayItem
当前
PlayItem
connection
_condition
将为0
IN_time
(到达时间)
OUT_time
(到达时间)
Clip1
TU映射时间轴
IN_time
(到达时间)
OUT_time
(到达时间)
Clip2
TU映射时间轴

图 31

句法	字节数	缩写
PlayItem(){		
Clip_information_file_name	8*10	bslbf
保留	24	bslbf
STC_sequence_id	8	uimsbf
IN_time	32	uimsbf
OUT_time	32	uimsbf
保留	14	bslbf
connection_condition	2	bslbf
if (<Virtual PlayList>){		
if (*connection_condition*=='*10*'){		
BridgeSequenceInfo()		
}		
}		
}		

图 32

PlayList() 中的CPI_type	IN_time的语义
EP_map类型	IN_time一定表示对应于PlayItem中的第一显示单元的33位长度的高32位
TU_map类型	IN_time一定是TU_map_time_axis上的时间，并一定四舍五入到time_unit精度。IN_time是由下述方程计算： $IN\_time = TU\_start\_time \%2^{32}$

图 33

PlayList() 中的CPI_type	OUT_time的语义
EP_map类型	OUT_time一定表示由下述方程计算的 Presentation_end_TS值的高32位: $Presentation\_end\_TS = PTS\_out + AU\_duration$ 这里PTS_out是对应于PlayItem中的最后显示单元的33位长的PTS。AU_duration是最后显示单元的90kHz显示时间。
TU_map类型	OUT_time一定是TU_map_time_axis上的时间，并四舍五入到time_unit精度。OUT_time由下式计算: $OUT\_time = TU\_start\_time \ \%2^{32}$

图 34

connection_condition	含义
00	.在先PlayItem对当前PlayItem的连接不保证为无缝重新播放. .如果PlayList的CPI_type是TU_map类型，该值一定设置在connection_condition中.
01	.该状态仅仅当PlayList的CPI_type为EP_map类型时是允许的. .因为系统时基的非连续点(STC基)，当前PlayItem和在先PlayItem表示分开.
10	.该状态仅仅当PlayList的CPI_type是EP_map类型时是允许的. .该状态仅对虚PlayList是允许的. .在先PlayItem对当前PlayItem的连接被保证为无缝重新播放. .在先PlayItem使用BridgeSequence连接到当前PlayItem. DVR MPEG-2传输流一定遵守后面说明的DVR-STD.
11	.该状态仅当PlayList的CPI_type是EP_map类型时是允许的. .在先PlayItem到当前PlayItem的连接保证为无缝重新播放. .在先PlayItem在不使用BridgeSequence的情况下连接到当前PlayItem. DVR MPEG-2传输流一定遵守后面说明的DVR-STD.

图 35

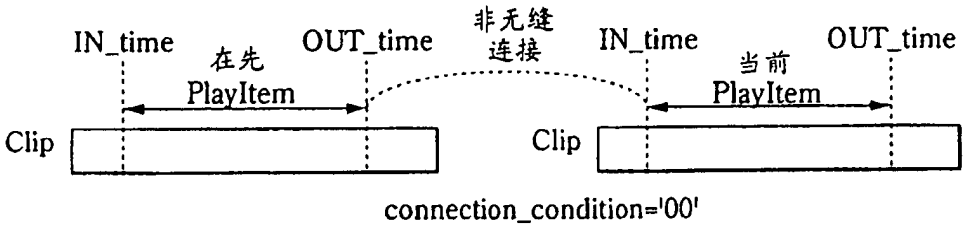


图 36A

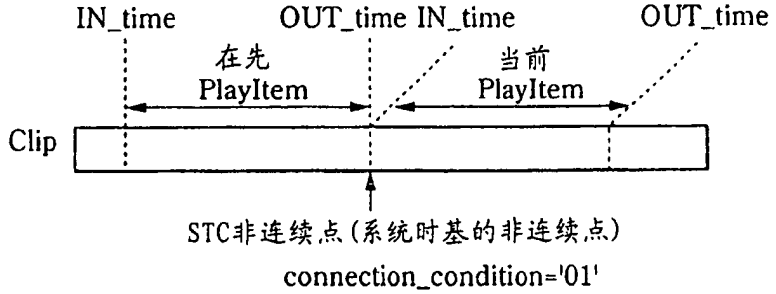


图 36B

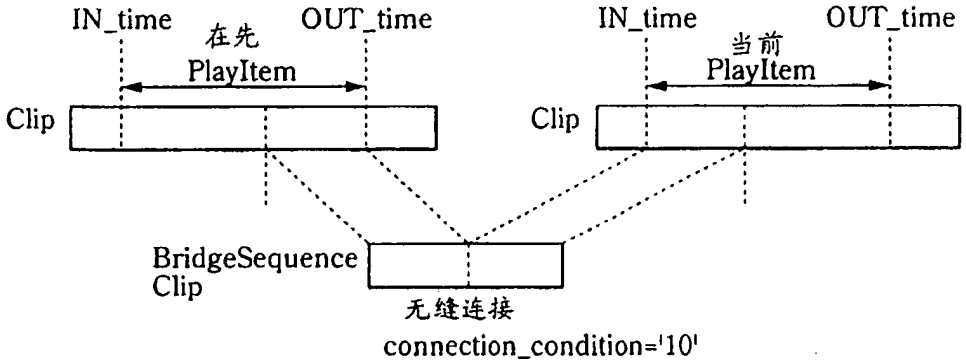


图 36C

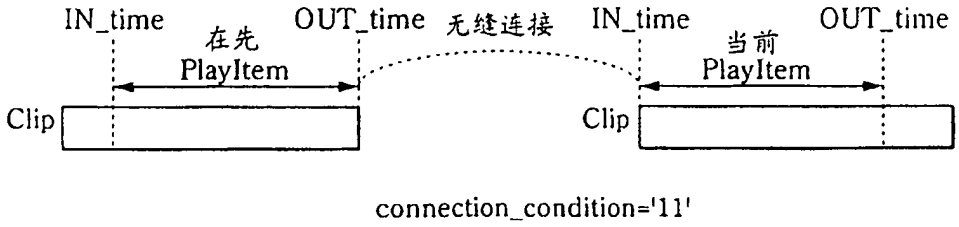


图 36D

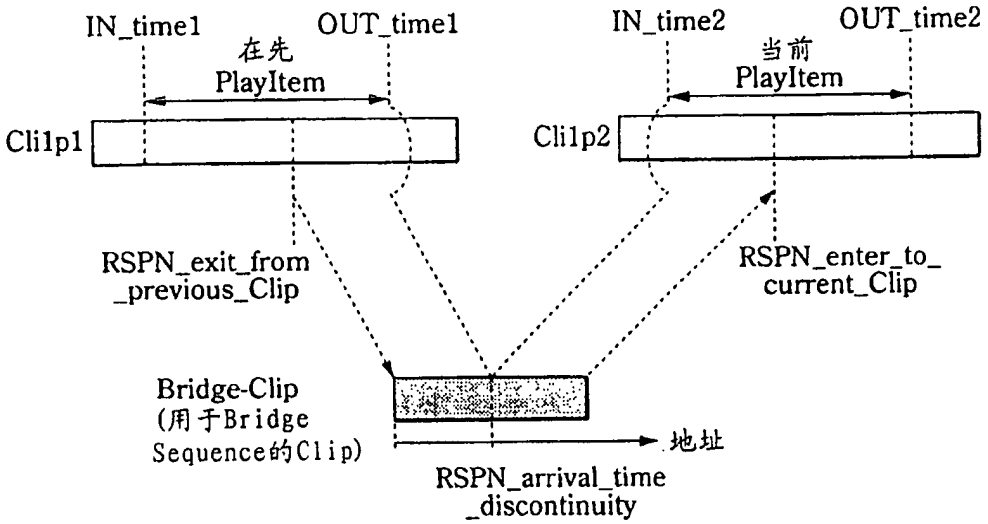

IN_time1
OUT_time1
在先
PlayItem
Clip1
IN_time2
OUT_time2
当前
PlayItem
Clip2
RSPN_exit_from
_previous_Clip
RSPN_enter_to_
current_Clip
Bridge-Clip
(用于Bridge
Sequence的Clip)
地址
RSPN_arrival_time
_discontinuity

图 37

句法	字节数	缩写
BridgeSequenceInfo() {		
Bridge_Clip_information_file_name	8*10	bslbf
RSPN_exit_from_previous_Clip	32	uimsbf
RSPN_enter_to_current_Clip	32	uimsbf
}		

图 38

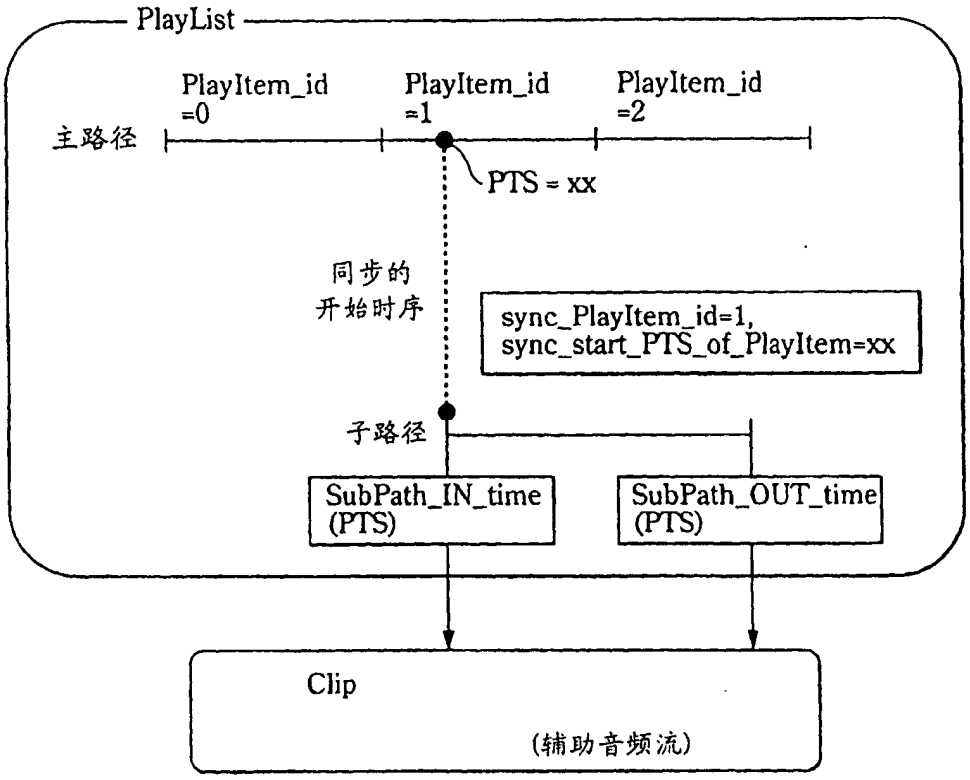

PlayList
PlayItem_id =0
PlayItem_id =1
PlayItem_id =2
主路径
PTS = xx
同步的开始时序
sync_PlayItem_id=1, sync_start_PTS_of_PlayItem=xx
子路径
SubPath_IN_time (PTS)
SubPath_OUT_time (PTS)
Clip
(辅助音频流)

图 39

句法	字节数	缩写
SubPlayItem(){		
Clip_Information_file_name	8*10	bslbf
SubPath_type	8	bslbf
sync_PlayItem_id	8	uimsbf
sync_start_PTS_of_PlayItem	32	uimsbf
SubPath_IN_time	32	uimsbf
SubPath_OUT_time	32	uimsbf
}		

图 40

SubPath_type	含义
0x00	辅助音频流路径
0x01-0xff	保留

图 41

句法	字节数	缩写
PlayListMark(){		
version_number	8*4	bslbf
length	32	uimsbf
number_of_PlayList_marks	16	uimsbf
for (i=0;i<*number_of_PlayList_marks*;i++){		
保留	8	bslbf
mark_type	8	bslbf
mark_time_stamp	32	uimsbf
PlayItem_id	8	uimsbf
保留	24	uimsbf
character_set	8	bslbf
name_length	8	uimsbf
mark_name	8*256	bslbf
ref_thumbnail_index	16	uimsbf
}		
}		

图 42

Mark_type	含义	说明
0x00	恢复-标记	重新播放恢复点。在PlayListMark()中定义的重新播放恢复点数一定为0或1。
0x01	书-标记	重新播放PlayList的入口点。该标记能够由用户设置和用作为指定喜爱的场景开始点的标记。
0x02	跳步-标记	跳步标记点。播放器从该点到程序结束跳过程序。在PlayListMark()中定义的跳步标记点数一定是0或1。
0x03-0x8F	保留	
0x90-0xFF	保留	保留用于ClipMark()

图 43

PlayList() 中的CPI_type	mark_time_stamp的语义
EP_map类型	mark_time_stamp一定表示对应于由标记 引用的表示单元的33位长度PTS的32位
TU_map类型	mark_time_stamp一定是关于TU_map_time_axis 的时间并且一定四舍五入到time_unit精度. mark_time_stamp通过下述方程计算: $mark\_time\_stamp = TU\_start\_time \%2^{32}$

图 44

句法	字节数	缩写
zzzzz.clpi {		
STC_Info_Start_address	32	uimsbf
ProgramInfo_Start_address	32	uimsbf
CPI_Start_address	32	uimsbf
ClipMark_Start_address	32	uimsbf
MakersPrivateData_Start_address	32	uimsbf
保留	96	bslbf
ClipInfo()		
for (i=0;i<N1;i++){		
padding_word	16	bslbf
}		
STC_Info()		
for (i=0;i<N2;i++){		
padding_word	16	bslbf
}		
ProgramInfo()		
for (i=0;i<N3;i++){		
padding_word	16	bslbf
}		
CPI()		
for (i=0;i<N4;i++){		
padding_word	16	bslbf
}		
ClipMark()		
for (i=0;i<N5;i++){		
padding_word	16	bslbf
}		
MakersPrivateData()		
}		

图 45

句法	字节数	缩写
ClipInfo(){		
version_number	8*4	bslbf
length	32	uimsbf
Clip_stream_type	8	bslbf
offset_SPN	32	uimsbf
TS_recording_rate	24	uimsbf
保留	8	bslbf
record_time_and_date	4*14	bslbf
保留	8	bslbf
duration	4*6	bslbf
保留	7	bslbf
time_controlled_flag	1	bslbf
TS_average_rate	24	uimsbf
if (Clip_stream_type==1) // Bridge-Clip AV stream		
RSPN_arrival_time_discontinuity	32	uimsbf
else		
保留	32	bslbf
reserved_for_system_use	144	bslbf
保留	11	bslbf
is_format_identifier_valid	1	bslbf
is_original_network_ID_valid	1	bslbf
is_transport_stream_ID_valid	1	bslbf
is_service_ID_valid	1	bslbf
is_country_code_valid	1	bslbf
format_identifier	32	bslbf
original_network_ID	16	uimsbf
transport_stream_ID	16	uimsbf
service_ID	16	uimsbf
country_code	24	bslbf
stream_format_name	16*8	bslbf
reserved_for_fortune_use	256	bslbf
}		

图 46

Clip_stream_type	含义
0	Clip AV流
1	Bridge-Clip AV流
2-255	保留

图 47

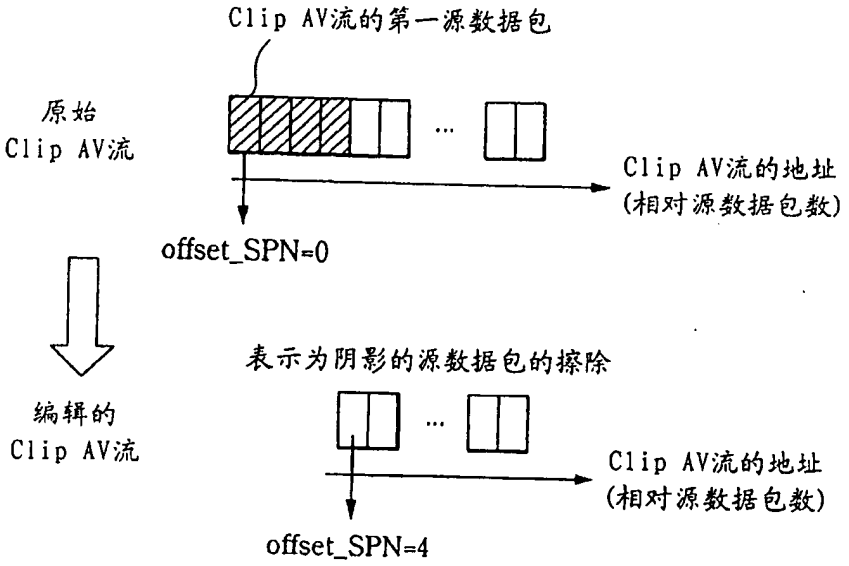

Clip AV流的第一源数据包
原始
Clip AV流
Clip AV流的地址
(相对源数据包数)
offset_SPN=0
表示为阴影的源数据包的擦除
编辑的
Clip AV流
Clip AV流的地址
(相对源数据包数)
offset_SPN=4

图 48

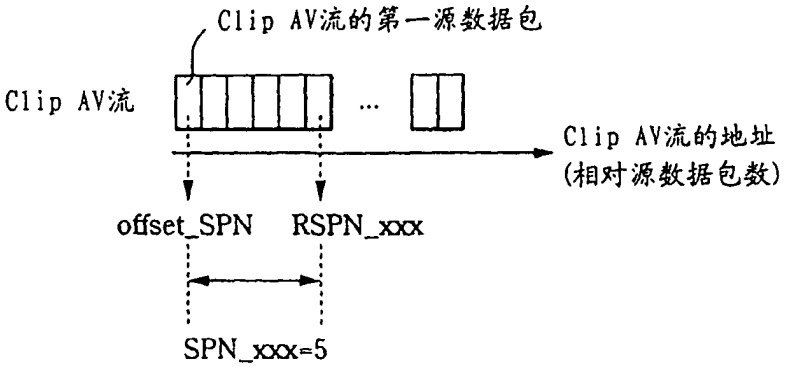


图 49

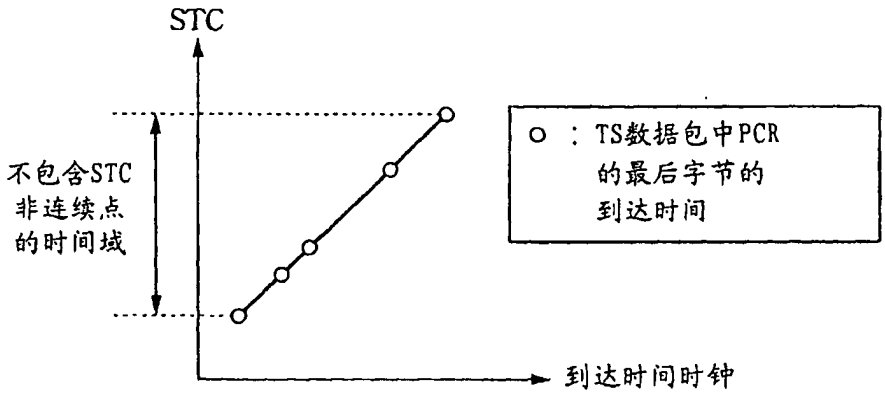


图 50A

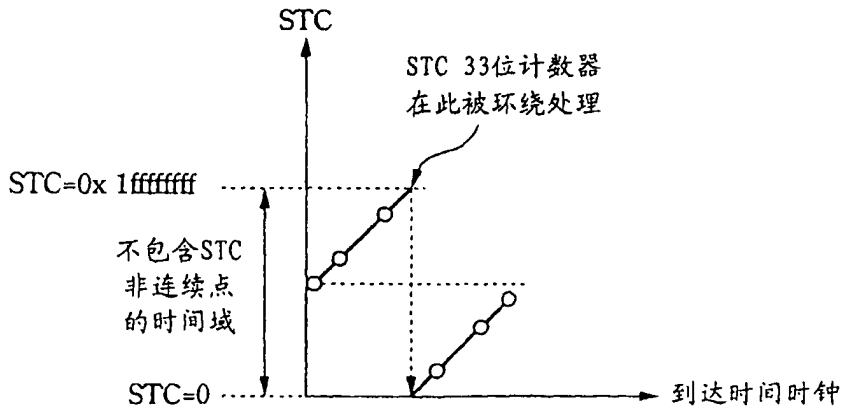


图 50B

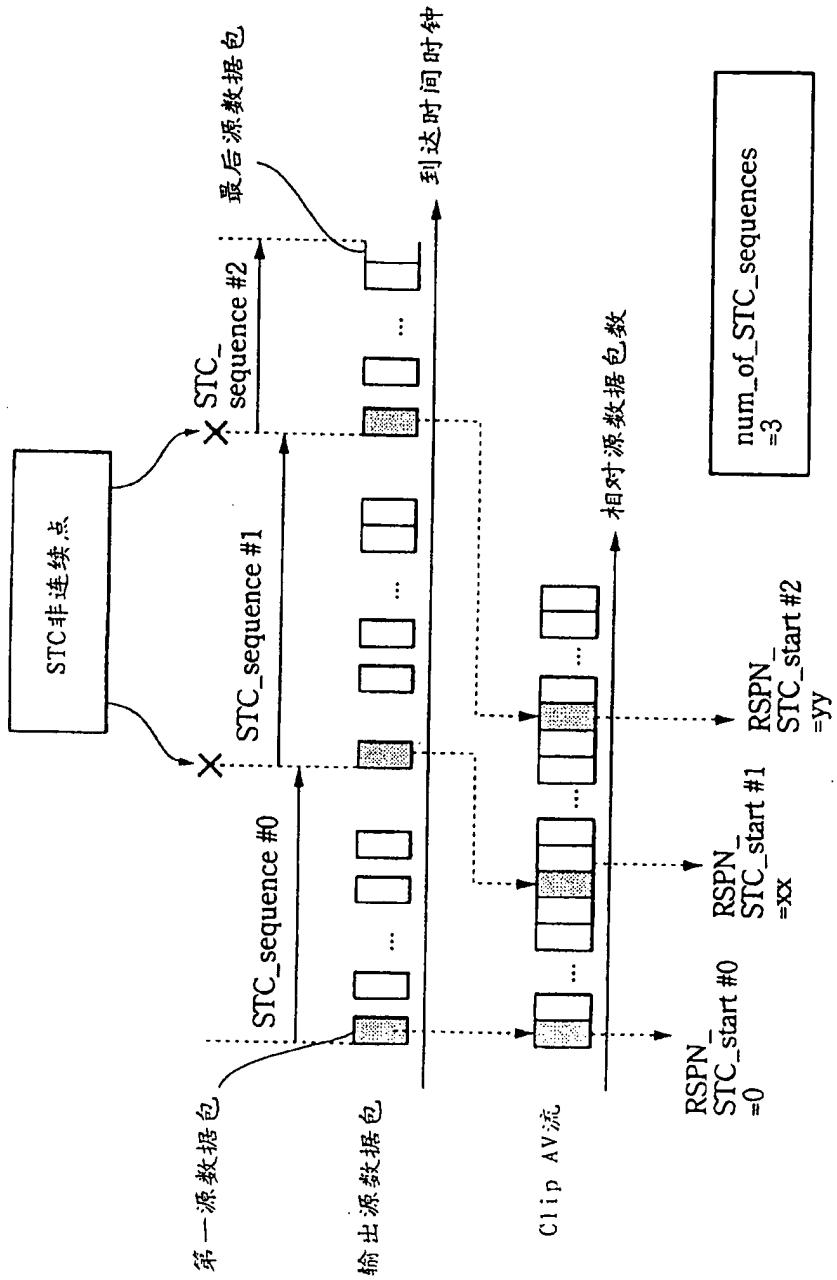

STC非连续点
STC_sequence #0
STC_sequence #1
STC_
sequence #2
最后源数据包
到达时间时钟
相对源数据包数
num_of_STC_sequences
=3
第一源数据包
输出源数据包
Clip AV流
RSPN_
STC_start #0
=0
RSPN_
STC_start #1
=xx
RSPN_
STC_start #2
=yy

图 51

句法	字节数	缩写
STC_Info(){		
version_number	8*4	bslbf
length	32	uimsbf
if (length !=0){		
保留	8	bslbf
num_of_STC_sequences	8	uimsbf
for (*STC_sequence_id=0;* *STC_sequence_id<num_of_STC_sequences;* *STC_sequence_id++*){		
保留	32	bslbf
RSPN_STC_start	32	uimsbf
}		
}		
}		

图 52

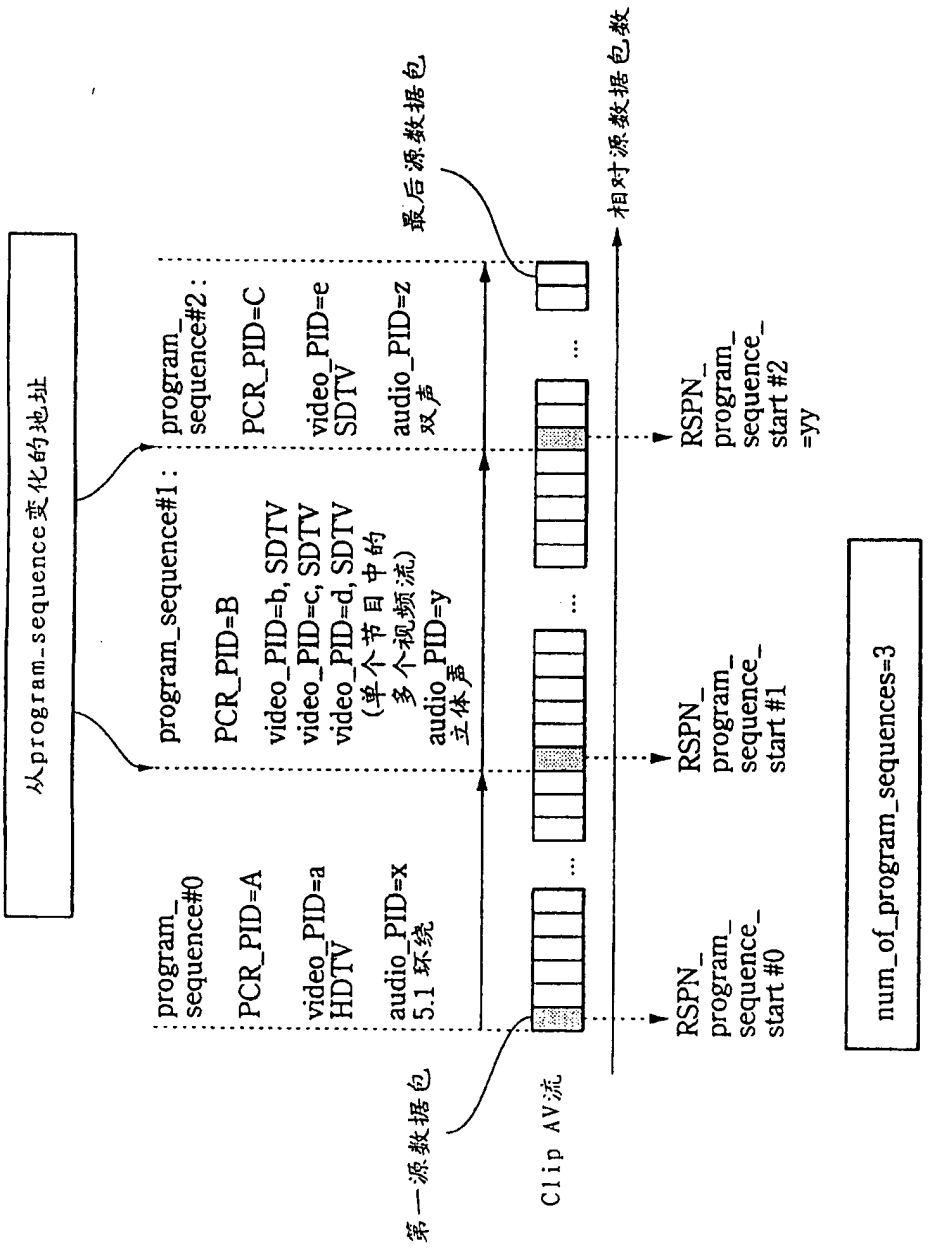

从program_sequence变化的地址
program_sequence#0
PCR_PID=A
video_PID=a HDTV
audio_PID=x 5.1环绕
program_sequence#1:
PCR_PID=B
video_PID=b, SDTV
video_PID=c, SDTV
video_PID=d, SDTV
(单个节目中的多个视频流)
audio_PID=y 立体声
program_sequence#2:
PCR_PID=C
video_PID=e SDTV
audio_PID=z 双声
第一源数据包
最后源数据包
Clip AV流
相对源数据包数
RSPN_program_sequence_start #0
RSPN_program_sequence_start #1
RSPN_program_sequence_start #2 =yy
num_of_program_sequences=3

图 53

句法	字节数	缩写
ProgramInfo(){		
version_number	8*4	bslbf
length	32	uimsbf
if (length !=0){		
保留	8	bslbf
number_of_program_sequences	8	uimsbf
for (i=0;i<*number_of_program_sequences*;i++){		
RSPN_program_sequence_start	32	uimsbf
保留	48	bslbf
PCR_PID	16	bslbf
number_of_videos	8	uimsbf
number_of_audios	8	uimsbf
for (k=0;k<*number_of_videos*;k++){		
video_stream_PID	16	bslbf
VideoCodingInfo()		
}		
for (k=0;k<*number_of_audios*;k++){		
audio_stream_PID	16	bslbf
AudioCodingInfo()		
}		
}		
}		
}		

图 54

句法	字节数	缩写
VideoCodingInfo() {		
video_format	8	uimsbf
frame_rate	8	uimsbf
display_aspect_ratio	8	uimsbf
保留	8	bslbf
}		

图 55

video_format	含义
0	480i
1	576i
2	480p (包括640 × 480p格式)
3	1080i
4	720p
5	1080p
6-254	保留
255	无信息

图 56

frame_rate	含义
0	禁止
1	24 000/1001 (23.976...)
2	24
3	25
4	30 000/1001 (29.97..)
5	30
6	50
7	60 000/1001 (59.94..)
8	60
9-254	保留
255	无信息

图 57

display_aspect_ratio	含义
0	禁止
1	保留
2	4: 3显示纵横比
3	16: 9显示纵横比
4-254	保留
255	无信息

图 58

句法	字节数	缩写
AudioCodingInfo() {		
audio_format	8	uimsbf
audio_component_type	8	uimsbf
sampling_frequency	8	uimsbf
保留	8	bslbf
}		

图 59

audio_coding	含义
0	MPEG-1音频层I或II
1	杜比AC-3音频
2	MPEG-2 AAC
3	MPEG-2多声道音频，与MPEG-1兼容的反向
4	SESF LPCM音频
5-254	保留
255	无信息

图 60

audio_component_type	含义
0	单声道
1	双声道
2	立体声(2声道)
3	多舌音，多声道
4	环绕声
5	视觉损坏的音频说明
6	难听的音频
7-254	保留
255	无信息

图 61

sampling_frequency	含义
0	48 kHz
1	44.1 kHz
2	32 kHz
3-254	保留
255	无信息

图 62

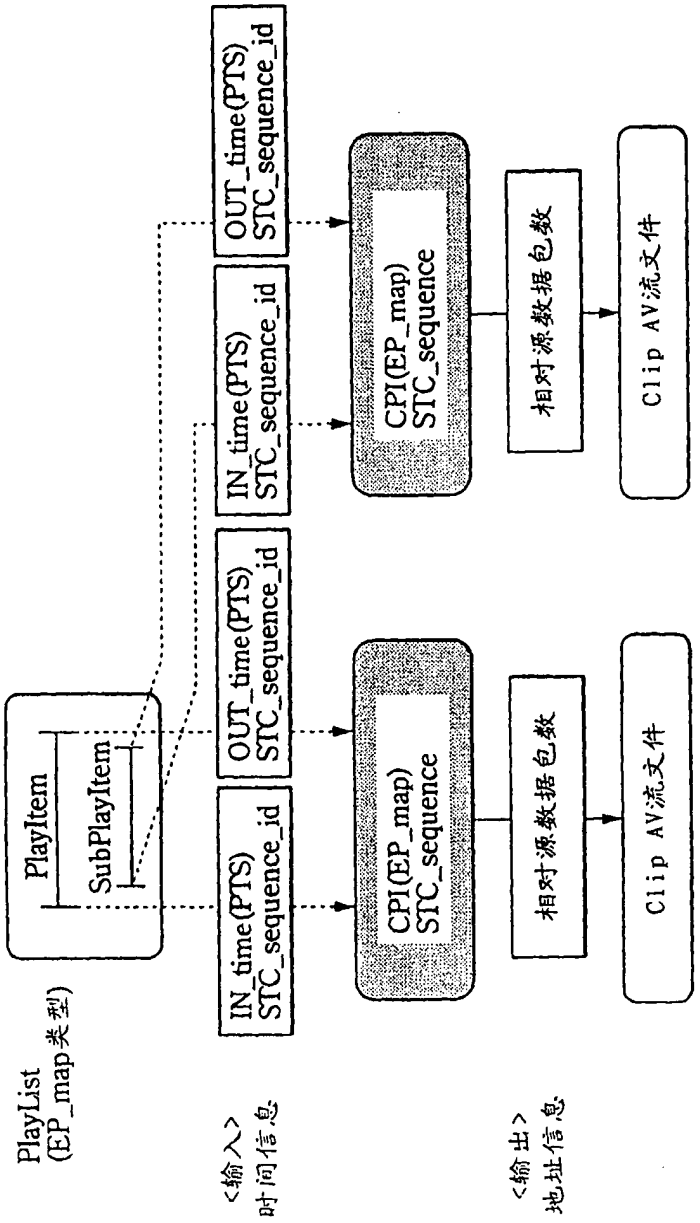

PlayList
(EP_map类型)
PlayItem
SubPlayItem
<输入>
时间信息
IN_time(PTS)
STC_sequence_id
OUT_time(PTS)
STC_sequence_id
CPI(EP_map)
STC_sequence
<输出>
地址信息
相对源数据包数
Clip AV流文件
IN_time(PTS)
STC_sequence_id
OUT_time(PTS)
STC_sequence_id
CPI(EP_map)
STC_sequence
相对源数据包数
Clip AV流文件

图 63

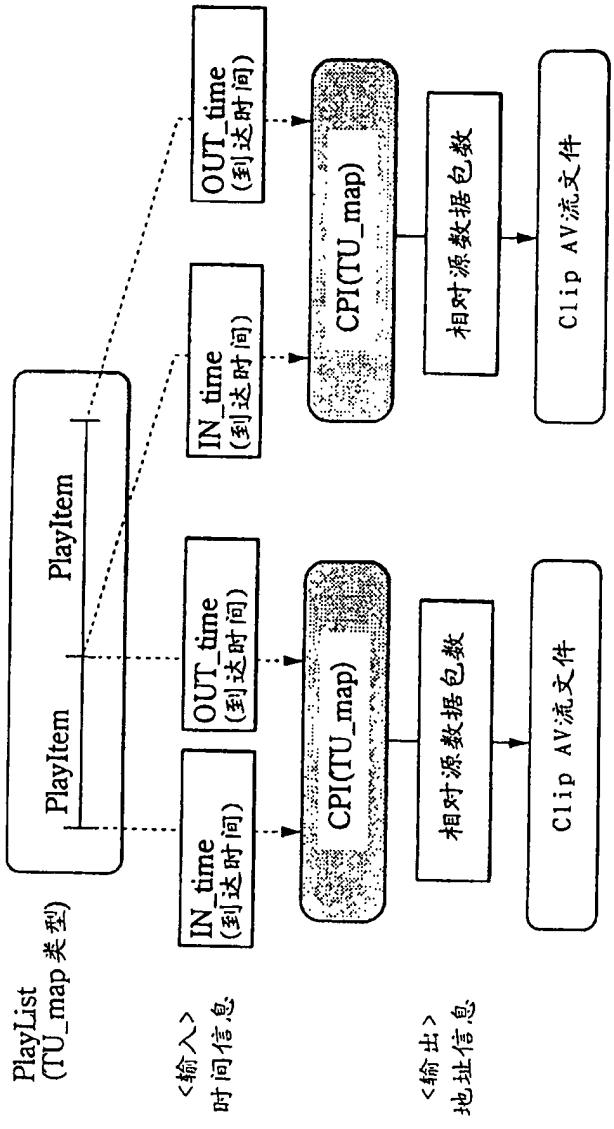

PlayList
(TU_map类型)
PlayItem
PlayItem
<输入>
时间信息
IN_time
(到达时间)
OUT_time
(到达时间)
IN_time
(到达时间)
OUT_time
(到达时间)
CPI(TU_map)
CPI(TU_map)
<输出>
地址信息
相对源数据包数
相对源数据包数
Clip AV流文件
Clip AV流文件

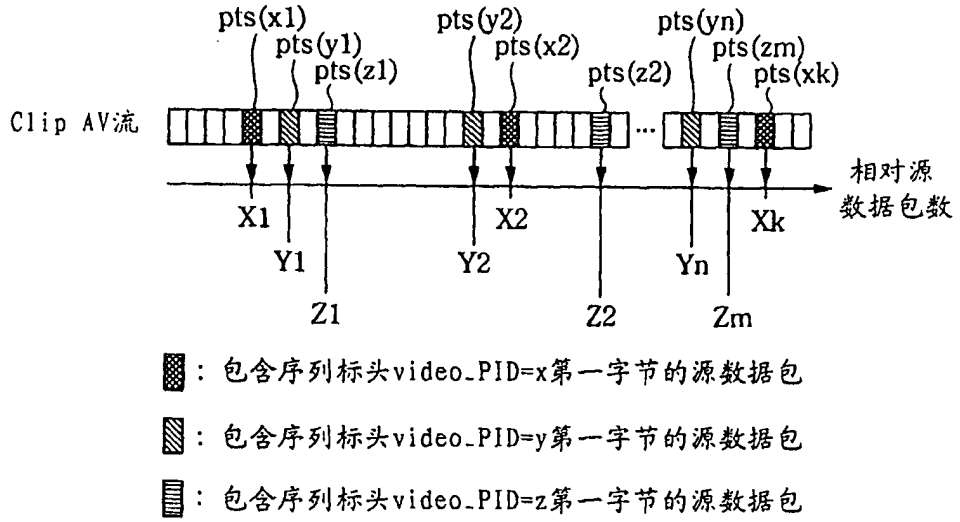
图 64

句法	字节数	缩写
CPI(){		
version_number	8*4	bslbf
length	32	uimsbf
保留	15	bslbf
CPI_type	1	bslbf
if (CPI_type==0)		
EP_map()		
else		
TU_map()		
}		

图 65

CPI_type	含义
0	EP映射类型
1	TU映射类型

图 66



EP_map

number_of_stream/PIDs=3

EP_map_for_ one_stream_PID(0)

stream_PID(0)=x
num_EP_entries(0)=k

PTS_EP start	RSPN_EP start
pts(x1)	X1
pts(x2)	X2
...	...
pts(xk)	Xk

EP_map_for_ one_stream_PID(1)

stream_PID(1)=y
num_EP_entries(1)=n

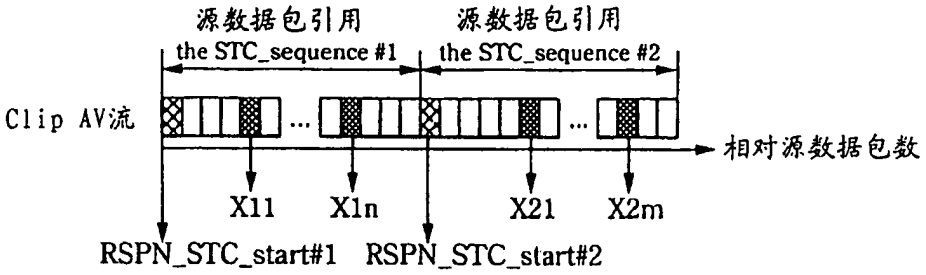
PTS_EP start	RSPN_EP start
pts(y1)	Y1
pts(y2)	Y2
...	...
pts(yn)	Yn

EP_map_for_ one_stream_PID(2)

stream_PID(2)=z
num_EP_entries(2)=m

PTS_EP start	RSPN_EP start
pts(z1)	Z1
pts(z2)	Z2
...	...
pts(zm)	Zm

图 67



▩：包含序列标头video_PID=x第一字节的源数据包

▩：由RSPN_STC_start(STC_into中定义的)引用的源数据包

EP_map_for_one_steram_ PID
video_PID=x

PTS_EP start	RSPN_EP start	
pts(x11)	X11	属于STC_sequence#1的数据
...	...	
pts(x1n)	X1n	
		边界
pts(x21)	X21	属于STC_sequence#2的数据
...	...	
pts(x2m)	X2m	

RSPN_STC_start #2<X21

图 68

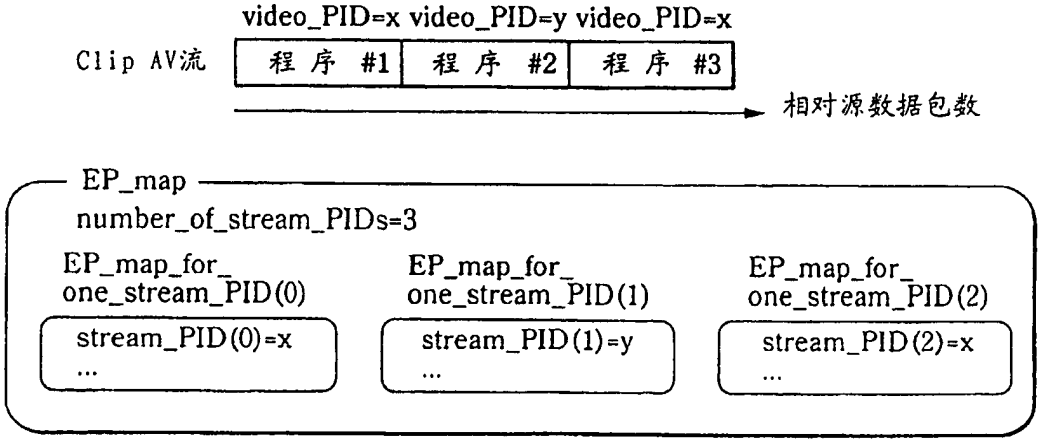


图 69

句法	字节数	缩写
EP_map(){		
保留	12	bslbf
EP_type	4	uimsbf
number_of_stream_PIDs	16	uimsbf
for (k=0;k<*number_of_stream_PIDs*;k++){		
stream_PID(*k*)	16	bslbf
num_EP_entries(*k*)	32	uimsbf
EP_map_for_one_stream_PID_Start_address(*k*)	32	uimsbf
}		
for (i=0;i<X;i++){		
padding_word	16	bslbf
}		
for (k=0;k<*number_of_stream_PIDs*;k++){		
EP_map_for_one_stream_PID(*num_EP_entries*(*k*))		
for (i=0;i<Y;i++){		
padding_word	16	bslbf
}		
}		
}		

图 70

EP_type	含义
0	视频
1	音频
2-15	保留

图 71

句法	字节数	缩写
EP_map_for_one_stream_PID(*N*) {		
for (i=0;i<*N*;i++){		
PTS_EP_start	32	uimsbf
RSPN_EP_start	32	uimsbf
}		
}		

图 72

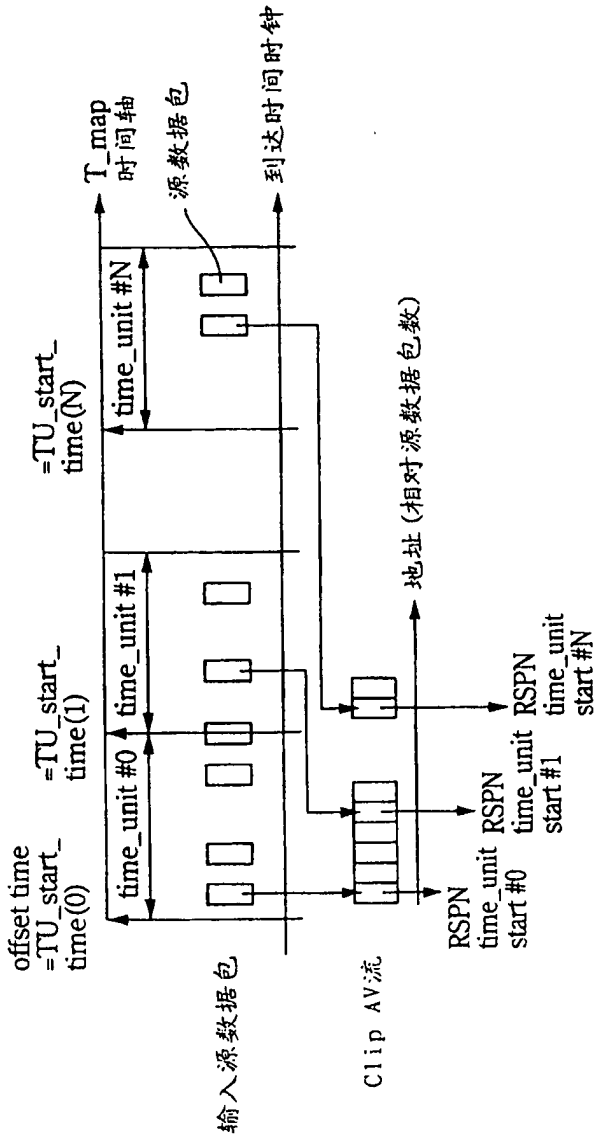

T_map
时间轴
源数据包
到达时间时钟
offset time
=TU_start_
time(0)
=TU_start_
time(1)
=TU_start_
time(N)
time_unit #0
time_unit #1
time_unit #N
地址 (相对源数据包数)
输入源数据包
Clip AV流
RSPN
time_unit
start #0
RSPN
time_unit
start #1
RSPN
time_unit
start #N

图 73

句法	字节数	缩写
TU_map() {		
offset_time	32	bslbf
time_unit_size	32	uimsbf
number_of_time_unit_entries	32	uimsbf
for (k=0;k<*number_of_time_unit_entries*;k++)		
RSPN_time_unit_start	32	uimsbf
}		

图 74

句法	字节数	缩写
ClipMark(){		
version_number	8*4	bslbf
length	32	uimsbf
number_of_Clip_marks	16	uimsbf
for (i=0; i<*number_of_clip_marks*; i++){		
保留	8	bslbf
mark_type	8	bslbf
mark_time_stamp	32	uimsbf
STC_sequence_id	8	uimsbf
保留	24	bslbf
character_set	8	bslbf
name_length	8	uimsbf
mark_name	8*256	bslbf
ref_thumbnail_index	16	uimsbf
}		
}		

图 75

Mark_type	含义	注释
0x00-0x8F	保留	保留用于PlayListMark()
0x90	事件开始标记	说明节目开始点的标记点
0x91	本地事件开始标记	说明节目中本地场景的标记点
0x92	场景开始标记	表示场景变化点的标记点
0x93-0xFF	保留	

图 76

PlayList() 中的CPI_type	mark_time_stamp的语义
EP_map类型	mark_time_stamp一定表示对应于由标记 引用的表示单元的33位长度PTS的高32位
TU_map类型	mark_time_stamp一定是TU_map_time_axis 上的时间并一定四舍五入到time_unit精度。 mark_time_stamp由下述方程计算: $mark\_time\_stamp = TU\_start\_time \%2^{32}$

图 77

句法	字节数	缩写
menu.thmb/mark.thmb {		
reserved	256	bslbf
Thumbnail()		
for (i=0; i<N1; i++)		
padding_word	16	bslbf
}		

图 78

句法	字节数	缩写
Thumbnail(){		
version_number	8*4	char
length	32	uimsbf
if (length !=0){		
tn_blocks_start_address	32	bslbf
number_of_thumbnails	16	uimsbf
tn_block_size	16	uimsbf
number_of_tn_blocks	16	uimsbf
reserved	16	bslbf
for (i=0; i<*number_of_thumbnails*; i++){		
thumbnail_index	16	uimsbf
thumbnail_picture_format	8	bslbf
reserved	8	bslbf
picture_data_size	32	uimsbf
start_tn_block_number	16	uimsbf
x_picture_length	16	uimsbf
y_picture_length	16	uimsbf
reserved	16	uimsbf
}		
stuffing_bytes	8*2*L1	bslbf
for(k=0; k<*number_of_tn_blocks*; k++){		
tn_block	tn_block_ size*1024*8	
}		
}		
}		

图 79

Thumbnail_picture_format	含义
0x00	MPEG-2视频I图像
0x01	DCF(受结束的JPEG)
0x02	PNG
0x03-0xff	保留

图 80

图 81A

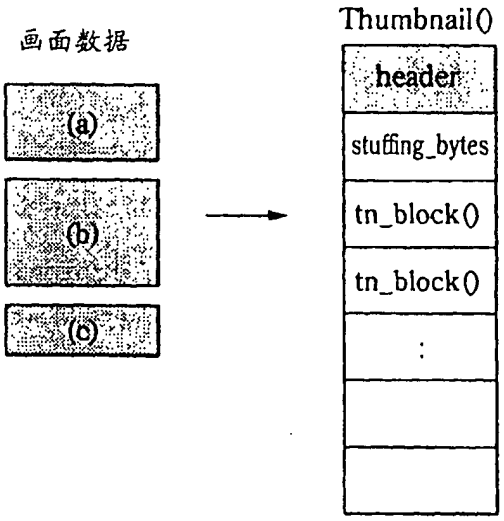
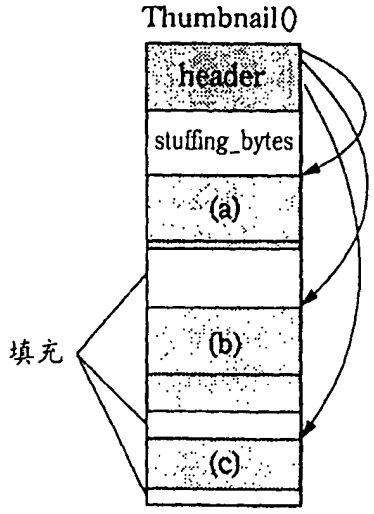


图 81B



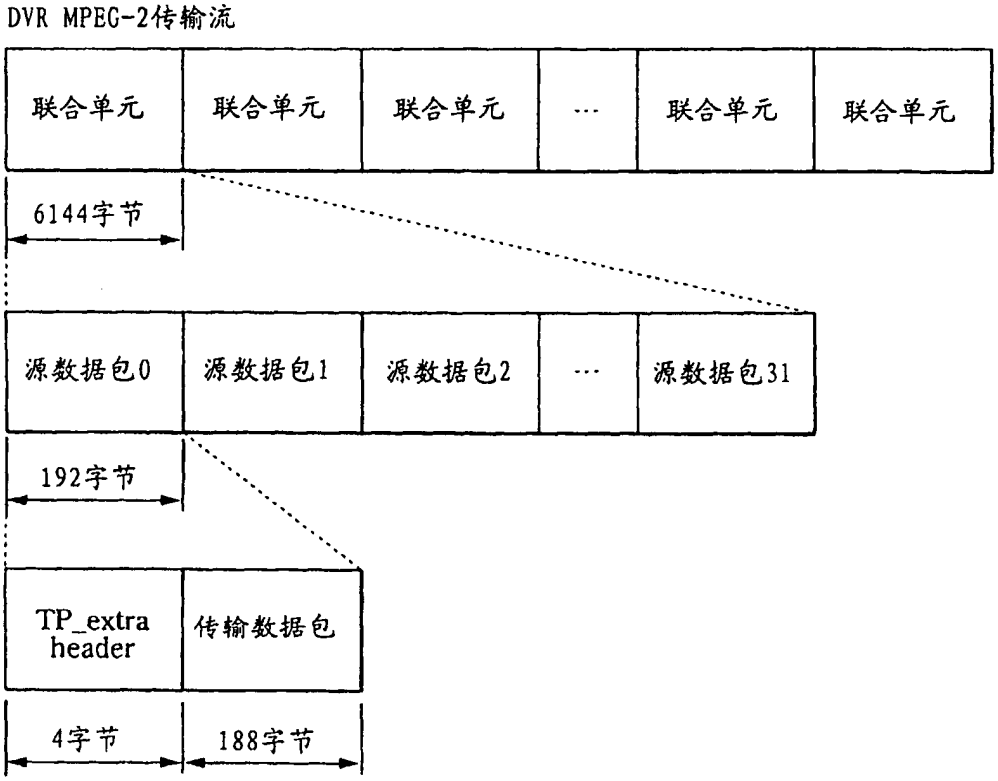

DVR MPEG-2传输流
联合单元
联合单元
联合单元
…
联合单元
联合单元
6144字节
源数据包0
源数据包1
源数据包2
…
源数据包31
192字节
TP_extra header
传输数据包
4字节
188字节

图 82

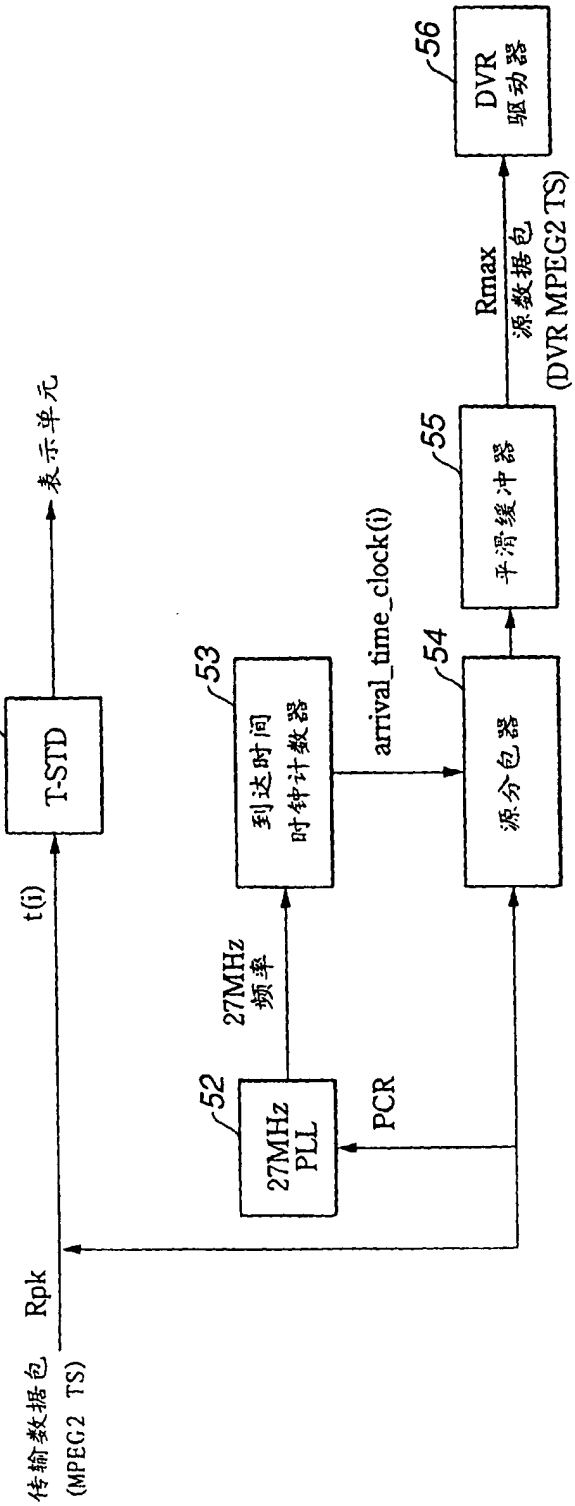

传输数据包
(MPEG2 TS)
Rpk
t(i)
51
T-STD
表示单元
52
27MHz
PLL
27MHz
频率
PCR
53
到达时间
时钟计数器
arrival_time_clock(i)
54
源分包器
55
平滑缓冲器
Rmax
源数据包
(DVR MPEG2 TS)
56
DVR
驱动器

图 83

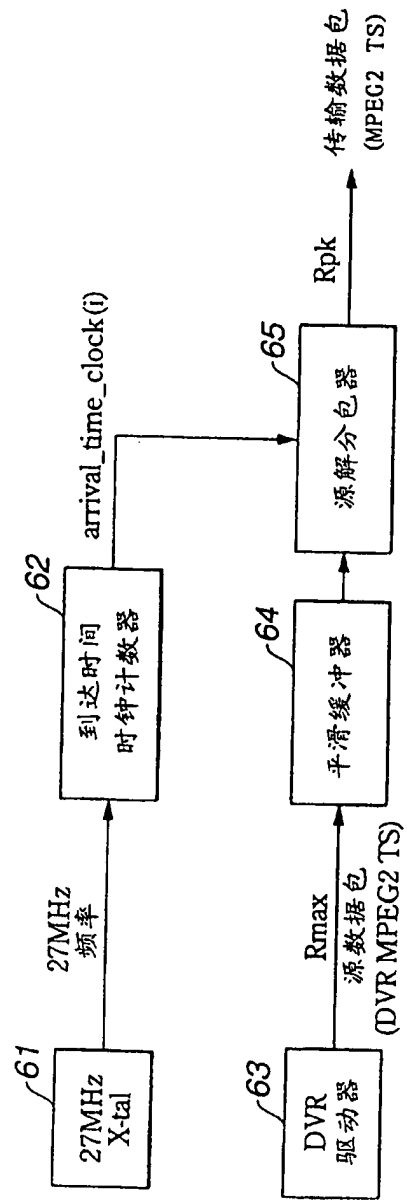

61
27MHz
X-tal
27MHz
频率
62
到达时间
时钟计数器
arrival_time_clock(i)
63
DVR
驱动器
Rmax
源数据包
(DVR MPEG2 TS)
64
平滑缓冲器
65
源解分包器
Rpk
传输数据包
(MPEG2 TS)

图 84

句法	字节数	缩写
source_packet() {		
TP_extra_header()		
transport_packet()		
}		

图 85

句法	字节数	缩写
TP_extra_header() {		
copy_permission_indicator	2	uimsbf
arrival_time_stamp	30	uimsbf
}		

图 86

copy_permission _indicator	含义
00	自由复制
01	不再复制
10	复制一次
11	禁止复制

图 87

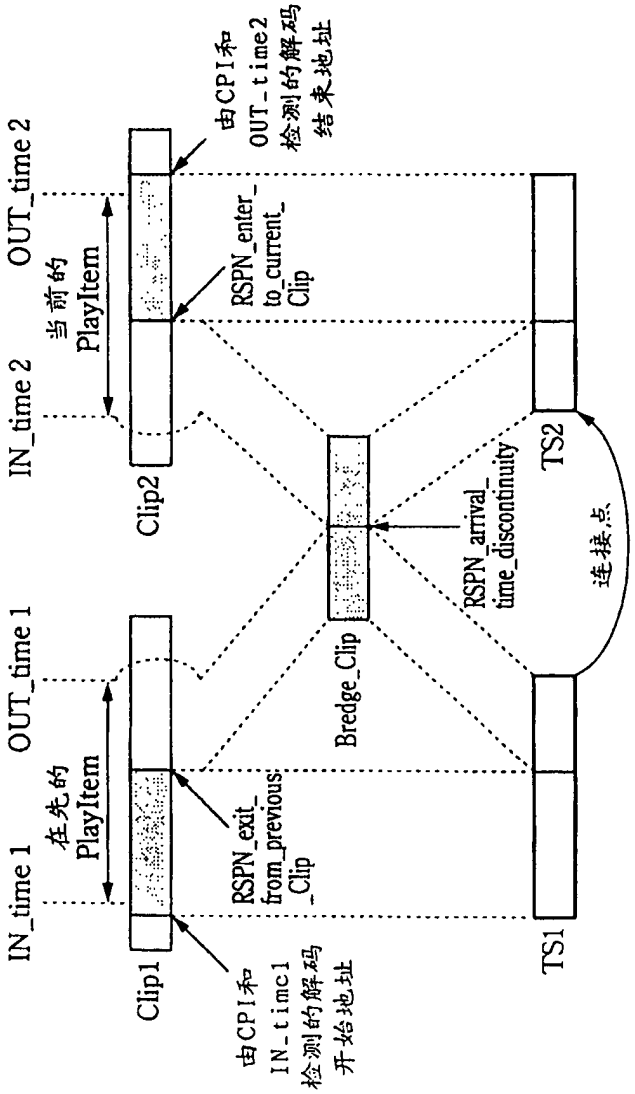

IN_time 1
OUT_time 1
在先的
PlayItem
RSPN_exit_
from_previous
_Clip
由CPI和
IN_timc1
检测的解码
开始地址
Clip1
IN_time 2
OUT_time 2
当前的
PlayItem
RSPN_enter_
to_current_
Clip
由CPI和
OUT_time2
检测的解码
结束地址
Clip2
Bredge_Clip
RSPN_arrival_
time_discontinuity
TS1
TS2
连接点

图 88

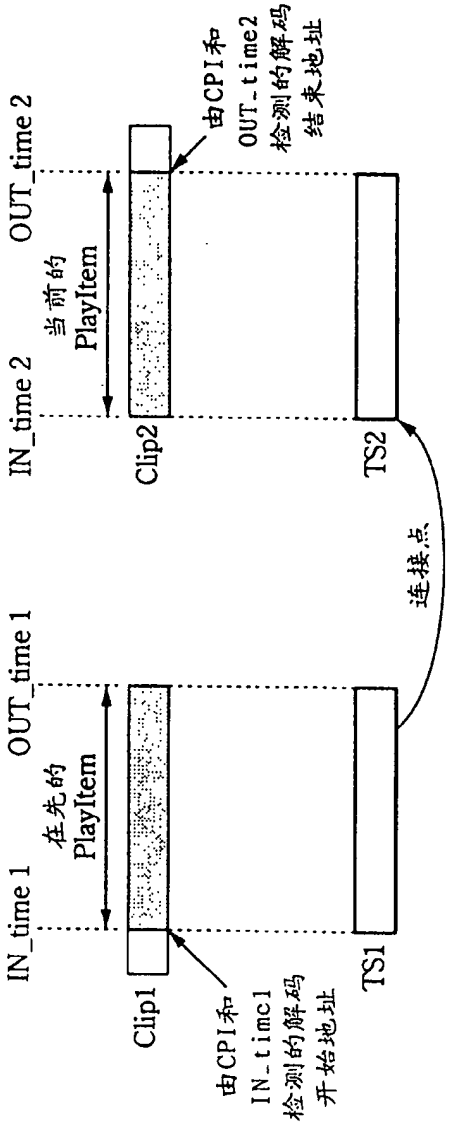

IN_time 1
OUT_time 1
在先的
PlayItem
Clip1
由CPI和
IN_timc1
检测的解码
开始地址
TS1
连接点
IN_time 2
OUT_time 2
当前的
PlayItem
Clip2
由CPI和
OUT_time2
检测的解码
结束地址
TS2

图 89

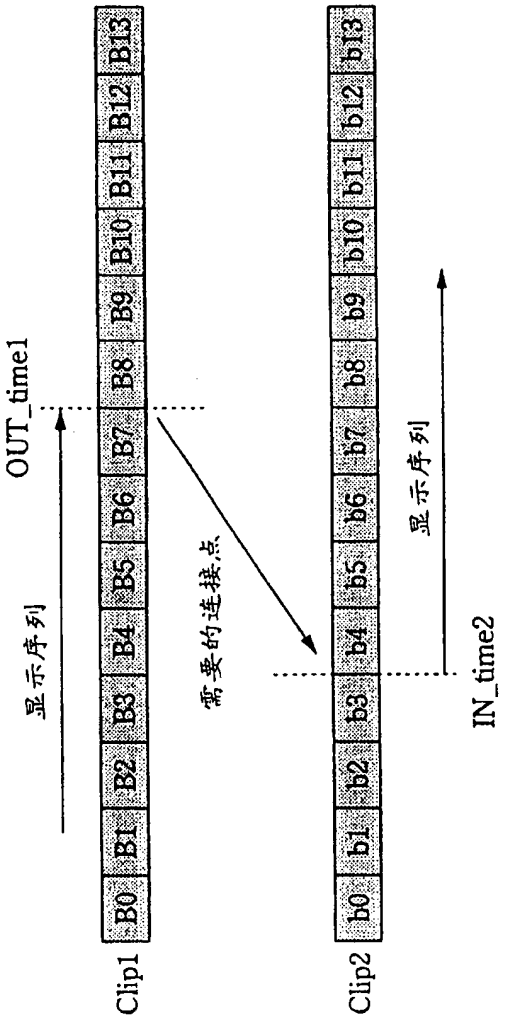

Clip1
B0
B1
B2
B3
B4
B5
B6
B7
B8
B9
B10
B11
B12
B13
OUT_time1
显示序列
需要的连接点
Clip2
b0
b1
b2
b3
b4
b5
b6
b7
b8
b9
b10
b11
b12
b13
显示序列
IN_time2

图 90

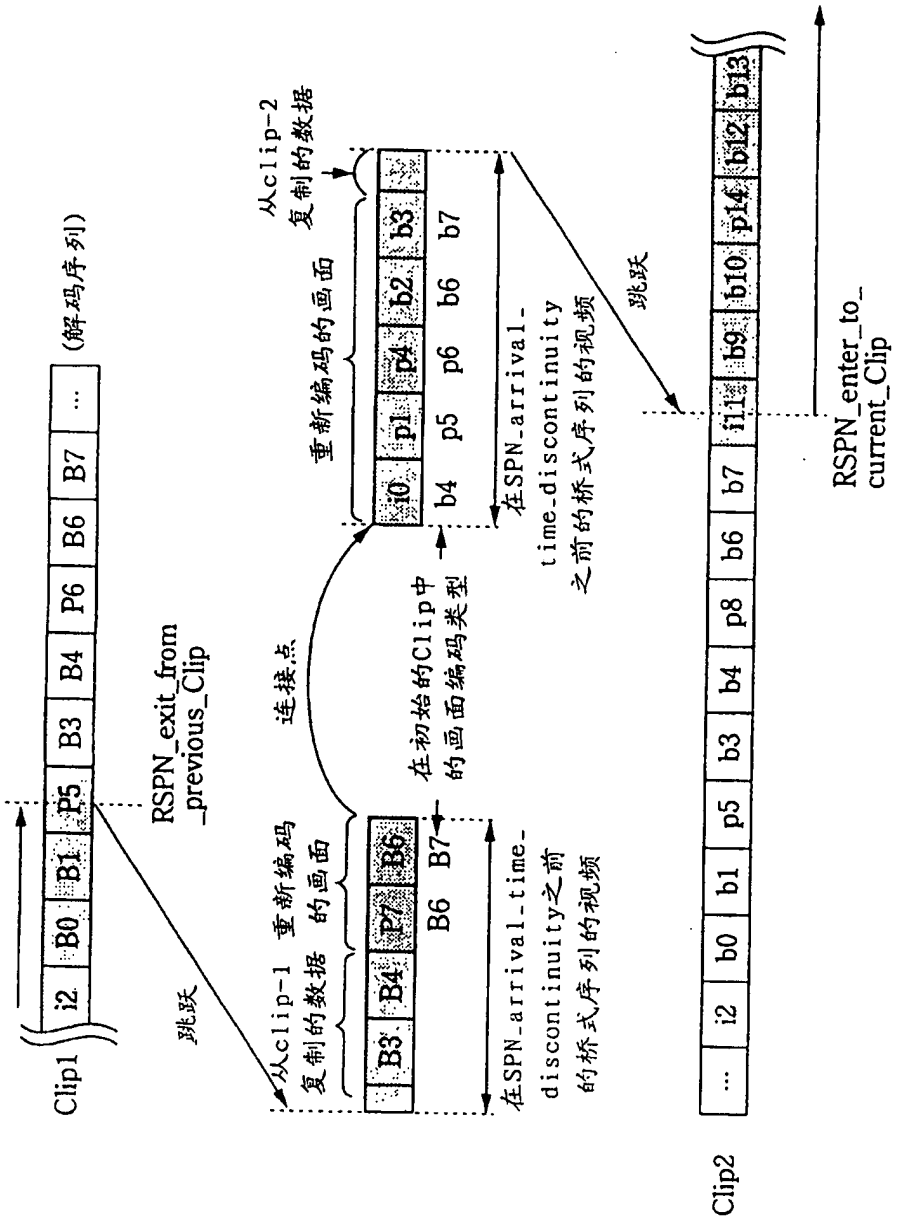

Clip1
i2
B0
B1
P5
B3
B4
P6
B6
B7
...
(解码序列)
RSPN_exit_from
_previous_Clip
跳跃
从clip-1 重新编码
复制的数据 的画面
连接点
从clip-2
复制的数据
重新编码的画面
在初始的Clip中
的画面编码类型
在SPN_arrival_time_
discontinuity之前
的桥式序列的视频
在SPN-arrival-
time-discontinuity
之前的桥式序列的视频
跳跃
RSPN_enter_to_
current_Clip
Clip2

图 91

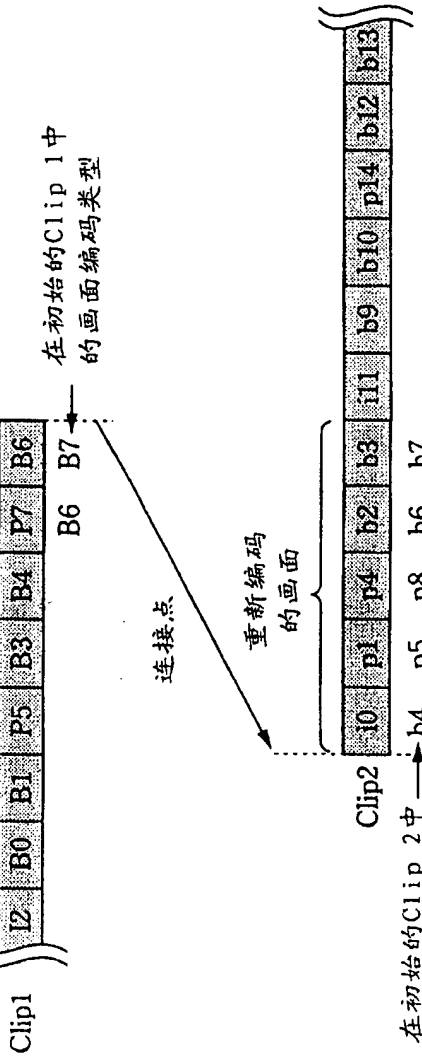

Clip1
(解码序列)
I2
B0
B1
P5
B3
B4
P7
B6
重新编码
的画面
B6
B7
在初始的Clip 1中
的画面编码类型
连接点
重新编码
的画面
Clip2
在初始的Clip 2中
的画面编码类型
i0
p1
p4
b2
b3
i11
b9
b10
p14
b12
b13
b4
p5
p8
b6
b7

图 92

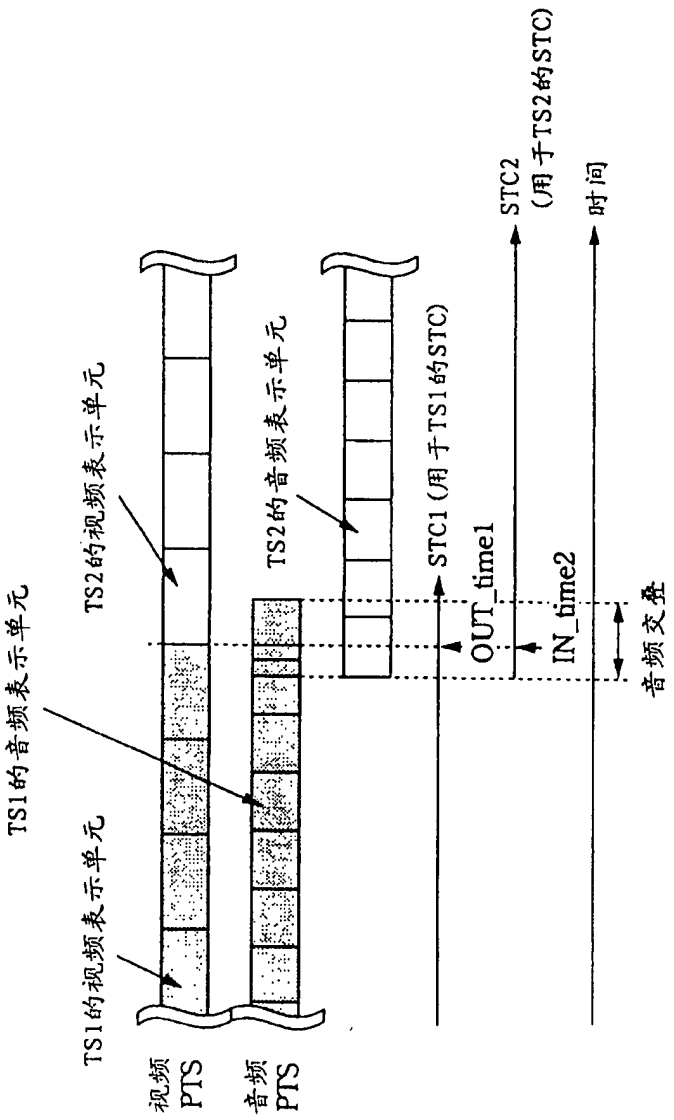

TS1的音频表示单元
TS1的视频表示单元
TS2的视频表示单元
TS2的音频表示单元
视频 PTS
音频 PTS
STC1(用于TS1的STC)
STC2 (用于TS2的STC)
时间
OUT_time1
IN_time2
音频交叠

图 93

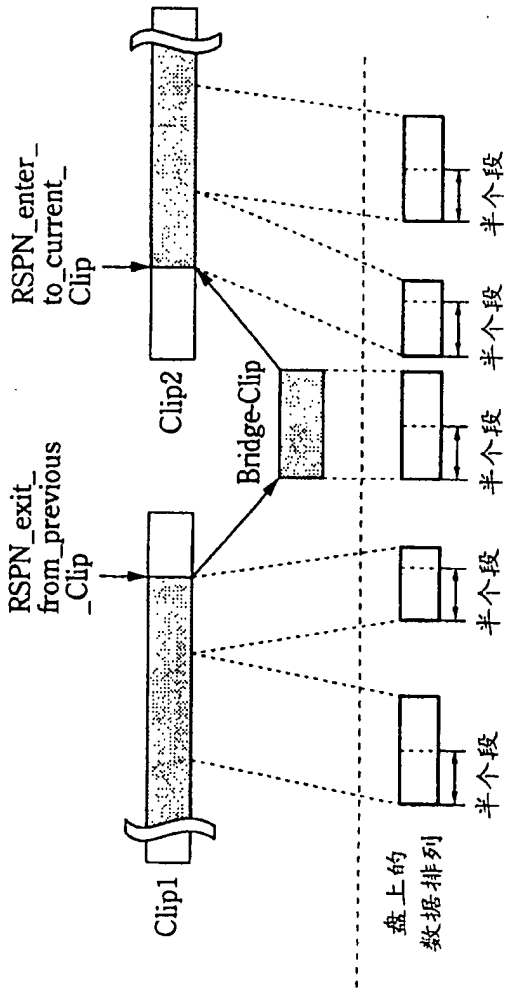

RSPN_enter_
to_current_
Clip
RSPN_exit_
from_previous
_Clip
Clip1
Clip2
Bridge-Clip
半个段
半个段
半个段
半个段
半个段
盘上的
数据排列

图 94

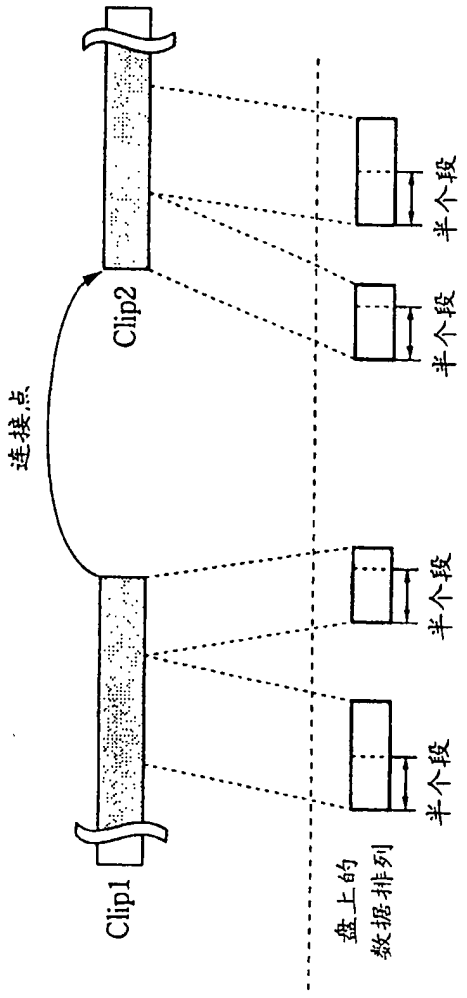


图 95

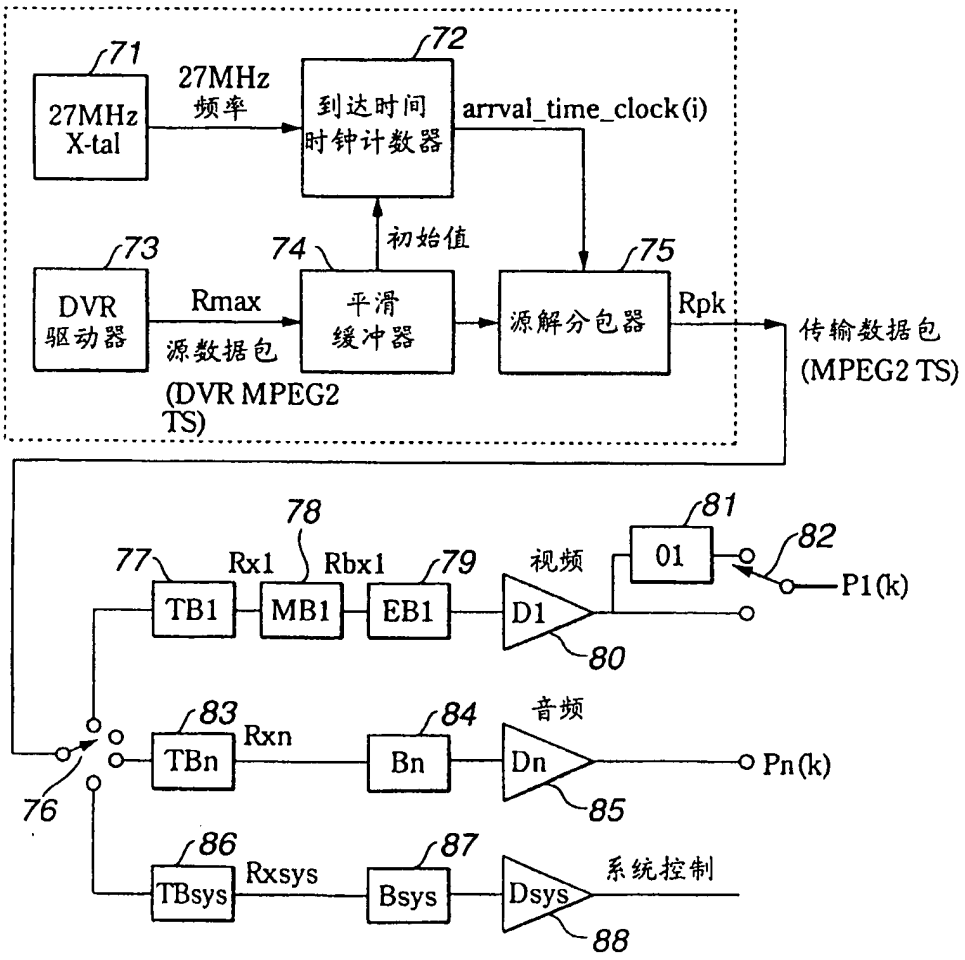

71
27MHz
X-tal
27MHz
频率
72
到达时间
时钟计数器
arrval_time_clock(i)
初始值
73
DVR
驱动器
Rmax
源数据包
(DVR MPEG2
TS)
74
平滑
缓冲器
75
源解分包器
Rpk
传输数据包
(MPEG2 TS)
78
77
Rx1
Rbx1
79
视频
81
01
82
P1(k)
TB1
MB1
EB1
D1
80
83
Rxn
84
音频
TBn
Bn
Dn
Pn(k)
85
76
86
Rxsys
87
系统控制
TBsys
Bsys
Dsys
88

图 96

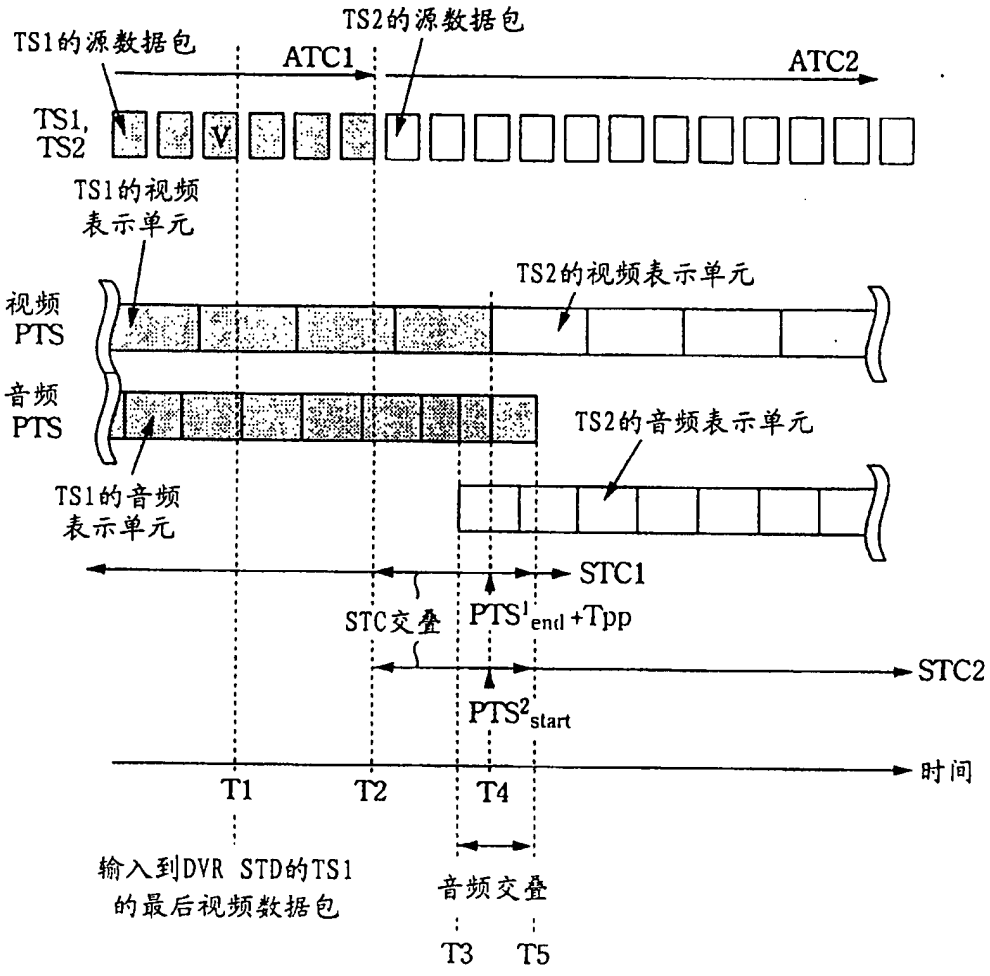

TS1的源数据包
TS2的源数据包
ATC1
ATC2
TS1,
TS2
V
TS1的视频
表示单元
TS2的视频表示单元
视频
PTS
音频
PTS
TS2的音频表示单元
TS1的音频
表示单元
STC1
STC交叠
PTS1end1+Tpp
STC2
PTS2start
时间
T1
T2
T4
输入到DVR STD的TS1
的最后视频数据包
音频交叠
T3
T5

图 97

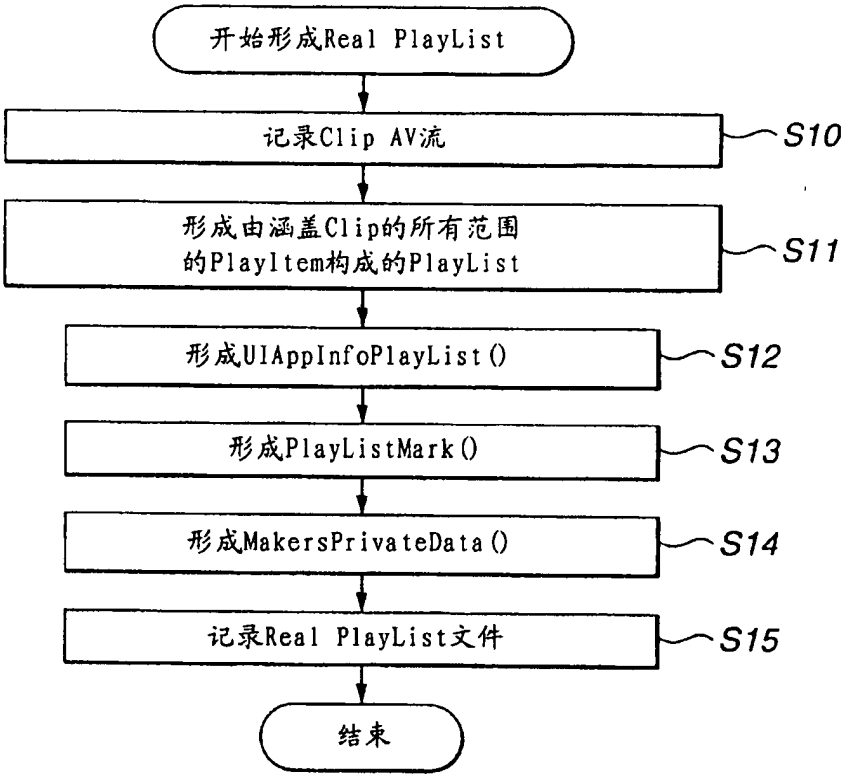

开始形成Real PlayList
记录Clip AV流
S10
形成由涵盖Clip的所有范围
的PlayItem构成的PlayList
S11
形成UIAppInfoPlayList()
S12
形成PlayListMark()
S13
形成MakersPrivateData()
S14
记录Real PlayList文件
S15
结束

图 98

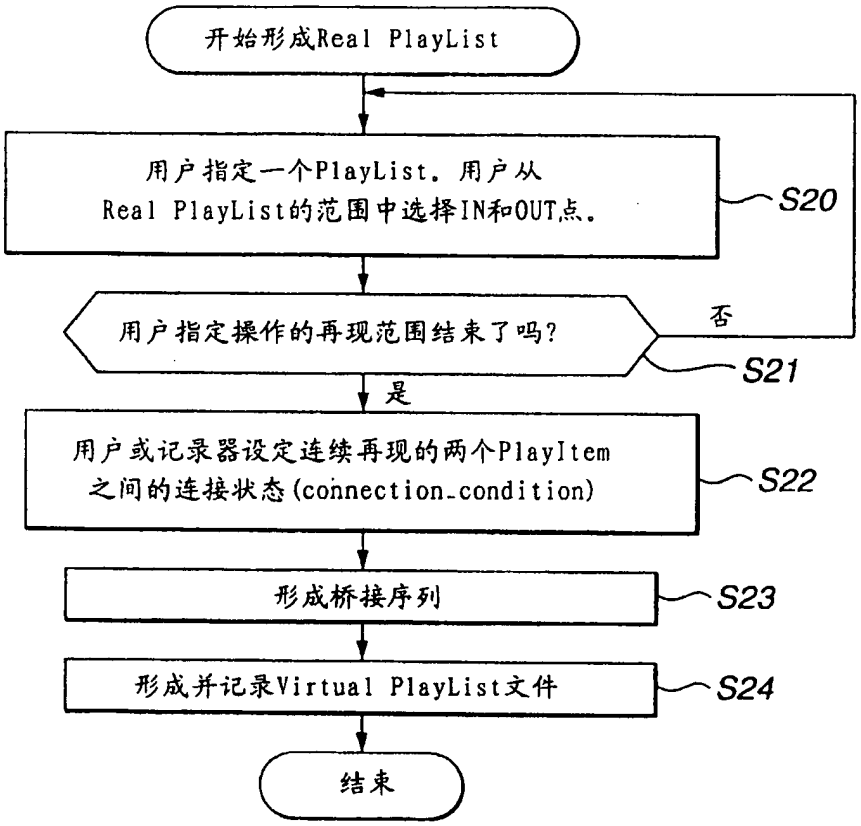

开始形成Real PlayList
用户指定一个PlayList。用户从
Real PlayList的范围中选择IN和OUT点。
S20
用户指定操作的再现范围结束了吗?
否
S21
是
用户或记录器设定连续再现的两个PlayItem
之间的连接状态(connection_condition)
S22
形成桥接序列
S23
形成并记录Virtual PlayList文件
S24
结束

图 99

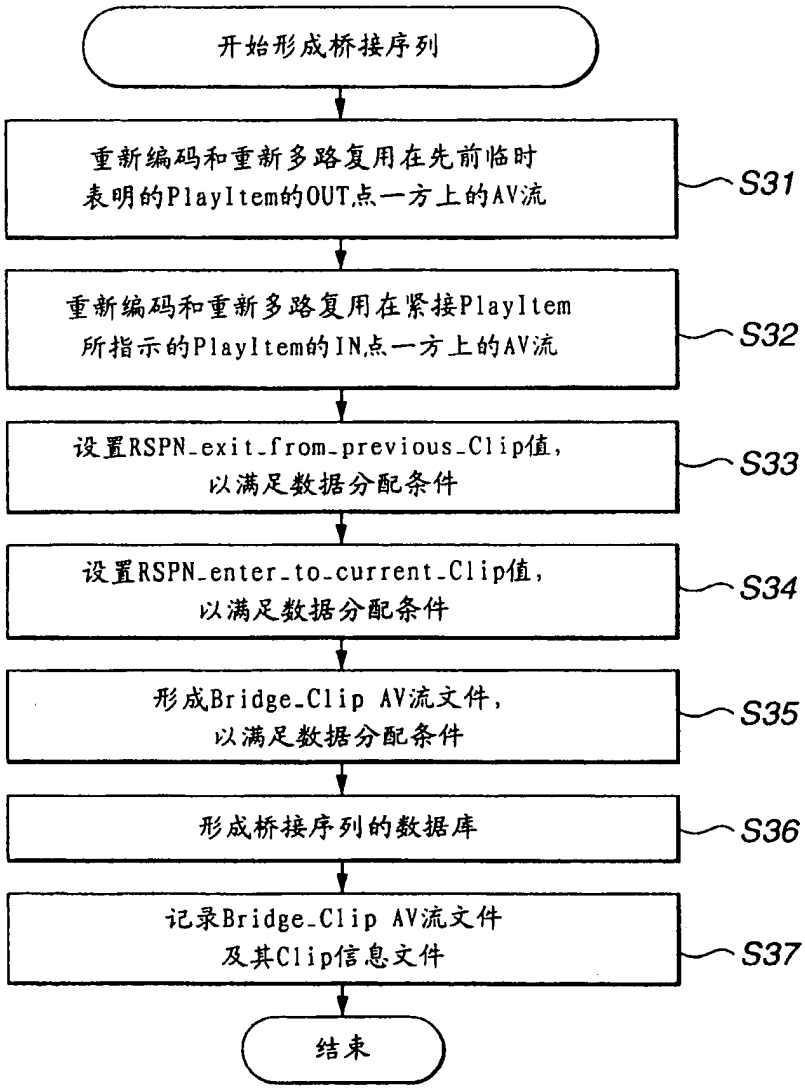

开始形成桥接序列
重新编码和重新多路复用在先前临时表明的PlayItem的OUT点一方上的AV流
S31
重新编码和重新多路复用在紧接PlayItem所指示的PlayItem的IN点一方上的AV流
S32
设置RSPN_exit_from_previous_Clip值，以满足数据分配条件
S33
设置RSPN_enter_to_current_Clip值，以满足数据分配条件
S34
形成Bridge_Clip AV流文件，以满足数据分配条件
S35
形成桥接序列的数据库
S36
记录Bridge_Clip AV流文件及其Clip信息文件
S37
结束

图 100

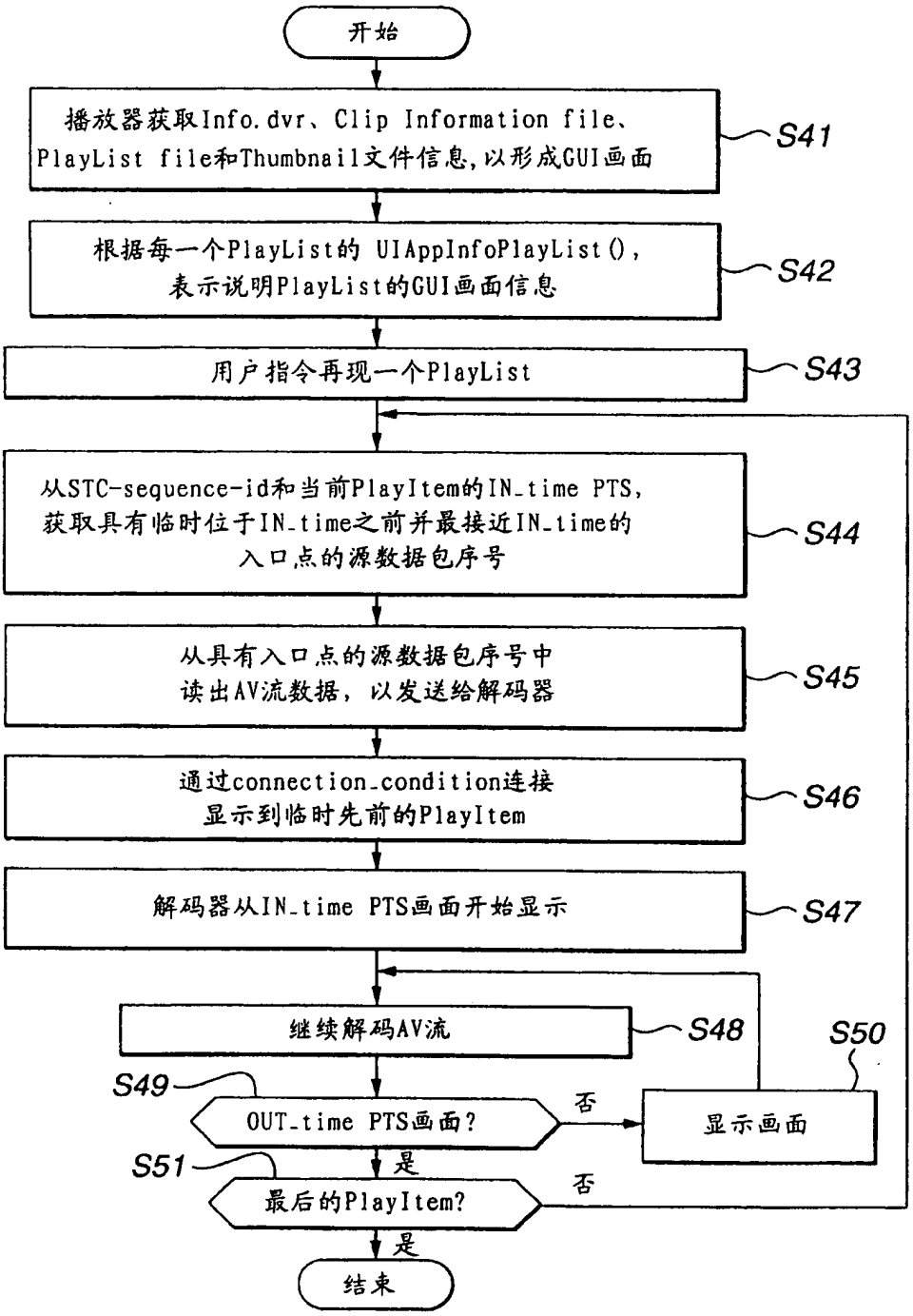

开始
播放器获取Info.dvr、Clip Information file、PlayList file和Thumbnail文件信息,以形成GUI画面
S41
根据每一个PlayList的 UIAppInfoPlayList(),表示说明PlayList的GUI画面信息
S42
用户指令再现一个PlayList
S43
从STC-sequence-id和当前PlayItem的IN_time PTS,获取具有临时位于IN_time之前并最接近IN_time的入口点的源数据包序号
S44
从具有入口点的源数据包序号中读出AV流数据,以发送给解码器
S45
通过connection_condition连接显示到临时先前的PlayItem
S46
解码器从IN_time PTS画面开始显示
S47
继续解码AV流
S48
S49
OUT_time PTS画面?
否
显示画面
S50
S51
是
最后的PlayItem?
否
是
结束

图 101

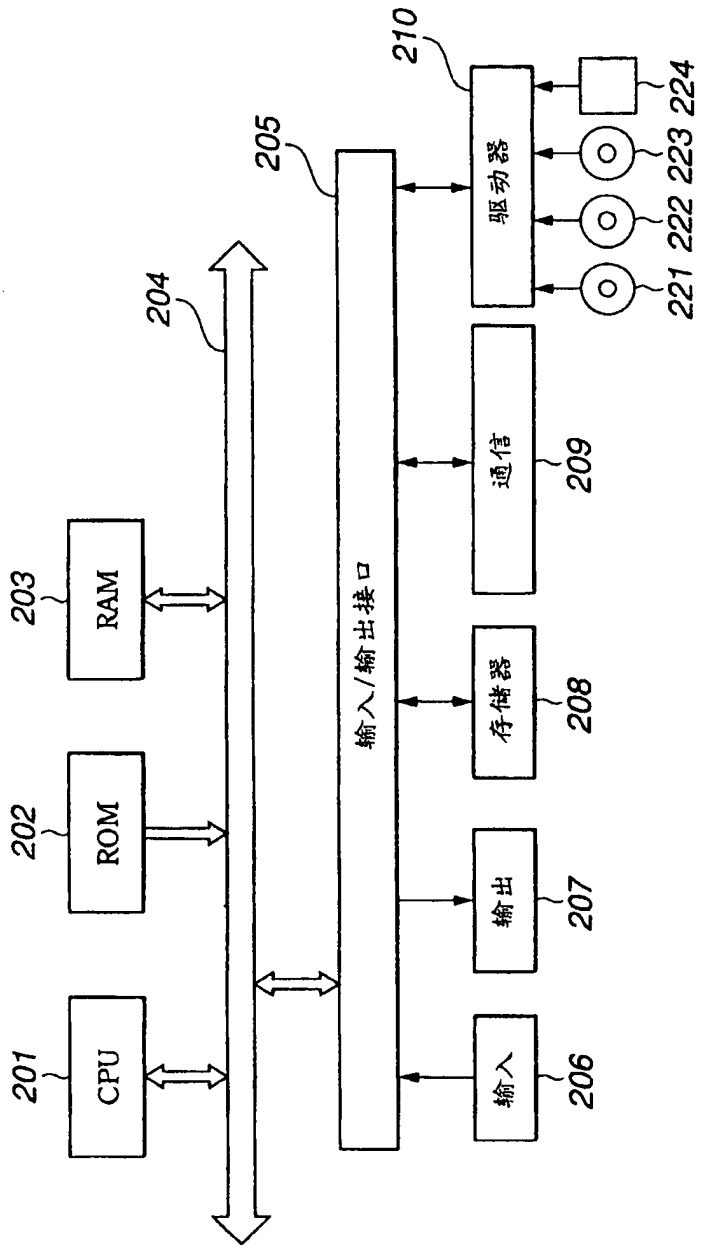

201
CPU
202
ROM
203
RAM
204
205
输入/输出接口
206
输入
207
输出
208
存储器
209
通信
210
驱动器
221
222
223
224

图 102